AMAN-DA: A knowledge reuse based approach for domain specific security requirements engineering

Amina Souag

▶ **To cite this version:**

Amina Souag. AMAN-DA: A knowledge reuse based approach for domain specific security requirements engineering. Other [cs.OH]. Université Paris 1 Panthéon Sorbonne, 2015. English. NNT : . tel-01302760

HAL Id: tel-01302760

https://hal-paris1.archives-ouvertes.fr/tel-01302760

Submitted on 15 Apr 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

THESE DE DOCTORAT

DE L'UNIVERSITE PARIS 1 PANTHEON-SORBONNE

Spécialité : Informatique

Présentée par :

Amina Souag

Pour l'obtention du grade de Docteur de

L'Université Paris 1 Panthéon – Sorbonne

AMAN-DA :

Une approche basée sur la réutilisation de la connaissance pour l'ingénierie des exigences de sécurité

Soutenue publiquement le 13 novembre 2015 devant la commission d'examen composée de :

Dr. Camille Salinesi. Professeur à l'Université Paris 1 Panthéon– Sorbonne, France.	Directeur de thèse
Dr. Isabelle Comyn Wattiau. Professeur au Conservatoire National des Arts et Métiers, France.	Co-directeur de thèse
Dr. Haris Mouratidis. Professeur à l'université de Brighton, Royaume Uni.	Rapporteur
Dr. Jean Donio. Professeur à l'université Paris 2 Panthéon-Assas, France.	Rapporteur
Dr. Raul Mazo. Maître de conferences à l'Université Paris 1 Panthéon –Sorbonne, France.	Examinateur

To Lameen, my eternal love

To Zachariah, our gift from God

To my parents Bachir and Nacera, for their unconditional support

To my aunt Rafika, for her generosity

"We must have perseverance and above all confidence in ourselves. We must believe that we are gifted for something and that this thing must be attained." Marie Curie

Une approche basée sur la réutilisation de la connaissance pour l'ingénierie des exigences de sécurité

Résumé

Au cours de ces dernières années, la sécurité des Systèmes d'Information (SI) est devenue une préoccupation importante, qui doit être prise en compte dans toutes les phases du développement du SI, y compris dans la phase initiale de l'ingénierie des exigences (IE). Prendre en considération la sécurité durant les premieres phases du dévelopment des SI permet aux développeurs d'envisager les menaces, leurs conséquences et les contre-mesures avant qu'un système soit mis en place. Les exigences de sécurité sont connues pour être "les plus difficiles des types d'exigences", et potentiellement celles qui causent le plus de risque si elles ne sont pas correctes. De plus, les ingénieurs en exigences ne sont pas principalement intéressés à, ou formés sur la sécurité. Leur connaissance tacite de la sécurité et leur connaissance primitive sur le domaine pour lequel ils élucident des exigences de sécurité rendent les exigences de sécurité résultantes pauvres et trop génériques.

Cette thèse explore l'approche de l'élucidation des exigences fondée sur la réutilisation de connaissances explicites. Tout d'abord, la thèse propose une étude cartographique systématique et exhaustive de la littérature sur la réutilisation des connaissances dans l'ingénierie des exigences de sécurité identifiant les diférentes formes de connaissances. Suivi par un examen et une classification des ontologies de sécurité comme étant la principale forme de réutilisation.

Dans la deuxième partie, AMAN-DA est présentée. AMAN-DA est la méthode développée dans cette thèse. Elle permet l'élucidation des exigences de sécurité d'un système d'information spécifique à un domaine particulier en réutilisant des connaissances encapsulées dans des ontologies de domaine et de sécurité. En outre, la thèse présente les différents éléments d'AMAN-DA : (i) une ontologie de sécurité noyau, (ii) une ontologie de domaine multi-niveau, (iii) des modèles syntaxique de buts et d'exigences de sécurité, (iv) un ensemble de règles et de mécanismes nécessaires d'explorer et de réutiliser la connaissance encapsulée dans les ontologies et de produire des spécifications d'exigences de sécurité.

La dernière partie rapporte l'évaluation de la méthode. AMAN-DA a été implémenté dans un prototype d'outil. Sa faisabilité a été évaluée et appliquée dans les études de cas de trois domaines différents (maritimes, applications web, et de vente). La facilité d'utilisation et l'utilisabilité de la méthode et de son outil ont également été évaluées dans une expérience contrôlée. L'expérience a révélé que la méthode est bénéfique pour l'élucidation des exigences de sécurité spécifiques aux domaines, et l'outil convivial et facile à utiliser.

Mots clés: éxigences de séccurité, ontologies, sécurité, étude cartographique, élucidation, ingénierie des exigences, domaine.

Centre de Recherche en Informatique

UFR 27
Centre de recherche en Informatique
Centre Pierre Mendès France
14ème étage, bureau C 14-03
90 rue de Tolbiac
75013 Paris

A knowledge reuse based approach for domain specific security requirements engineering

Abstract

In recent years, security in Information Systems (IS) has become an important issue that needs to be taken into account in all stages of IS development, including the early phase of Requirement Engineering (RE). Considering security during early stages of IS development allows IS developers to envisage threats, their consequences and countermeasures before a system is in place. Security requirements are known to be "the most difficult of requirements types", and potentially the ones causing the greatest risk if they are not correct. Moreover, requirements engineers are not primarily interested in, or knowledgeable about, security. Their tacit knowledge about security and their primitive knowledge about the domain for which they elicit security requirements make the resulting security requirements poor and too generic.

This thesis explores the approach of eliciting requirements based on the reuse of explicit knowledge. First, the thesis proposes an extensive systematic mapping study of the literature on the reuse of knowledge in security requirements engineering identifying the diferent knowledge forms. This is followed by a review and classification of security ontologies as the main reuse form.

In the second part, AMAN-DA is presented. AMAN-DA is the method developed in this thesis. It allows the elicitation of domain-specific security requirements of an information system by reusing knowledge encapsulated in domain and security ontologies. Besides that, the thesis presents the different elements of AMANDA: (i) a core security ontology, (ii) a multi-level domain ontology, (iii) security goals and requirements's syntactic models, (iv) a set of rules and mechanisms necessary to explore and reuse the encapsulated knowledge of the ontologies and produce security requirements specifications.

The last part reports the evaluation of the method. AMAN-DA was implemented in a prototype tool. Its feasibility was evaluated and applied in case studies of three different domains (maritime, web applications, and sales). The ease of use and the usability of the method and its tool were also evaluated in a controlled experiment. The experiment revealed that the method is beneficial for the elicitation of domain specific security requirements, and that the tool is friendly and easy to use.

Keywords: security requirements, ontologies, security, mapping study, elicitation, requirements engineering, domain.

Acknowledgments

I come from Dellys, a beautiful coastal city in northern Algeria. This thesis is the culmination of a long journey and the realisation of a dream I had. When I reached the University of Paris 1 Panthéon-Sorbonne five years ago, it was the beginning of a real adventure, sometimes difficult, but rich and constructive. The result wouldn't have been possible without the help and guidance of many people…

I would like to express my deepest gratitude to my first supervisor, Prof. Camille Salinesi, for the trust he has placed in me by welcoming me to the Centre de Recherche en Informatique and accepting the scientific supervision of my work. I thank him for his patient guidance, encouragement, knowledge and advice. I would also like to thank Prof. Isabelle Comyn-Wattiau, who co-supervised my thesis and patiently corrected my writing. Her suggestions helped me develop the ideas of this thesis.

Besides my advisors, I would like to thank Prof. Haris Mouratidis of the University of Brighton and Prof. Jean Donio of Paris 2 Panthéon-Assas University for agreeing to review my thesis, and for their insightful comments and encouragement – but also for their hard questions which encouraged me to widen my research from various perspectives.

I would like to thank Dr. Raul Mazo, Associate Professor at Paris 1 Panthéon-Sorbonne University, for accepting to be part of the defense committee. On top of that, Raul was always willing to help and give his best suggestions; without his valuable support it would not have been possible to conduct this research.

Many thanks to the (ex) lab members, in particular Prof. Bénédicte le Grand for the stimulating discussions and advice on my work which helped me improve the results, and Dr. Manuele Kirsch-Pinheiro who was very helpful in the organization of my teaching classes. Sana, Ali, Cosmin, Hela, Ghazaleh, Dany, Salma, Raouia, Farah, Juan Carlos, Abdelkader, Elena and Elena – thank you all!

I also thank my friends from the ISID-CNAM lab. I am so grateful to Zeinab Hmedeh for being such a good friend and for her help in developing the AMAN-DA tool, and to Feten, Lydia, Houda, Sarah, Ryadh, Pascal, Anh Do, of whom I will always have nice memories.

Last but not the least, I would like to thank my family: my parents, my sisters Selma and Sarah and my brother Houssam, thank you for believing in me, thanks for your patience and the affection you have shown me over the years. My family in law, thank you for your help and your great support. My cousin Rym, uncle Omar and his family, Baya, Mahdia and their families – thank you for your kindness.

Special thanks to my aunt Rafika and her family, this great and generous lady, without whom this project would never have started. She was the one who opened the first door.

Lameen, my husband, and my beloved, thank you for your help, your patience, and your great support. I will always remember the day you said: "I want you to do this PhD and I will always be besides you". And you were.

Zachariah, you are only ten months while I am writing those lines, but you have already been such a nice and patient baby. Thank you my little angel.

والحمد والشكر لله رب العالمين

CONTENTS

Part 1: State of the art

CHAPTER 2
Security requirements engineering and knowledge reuse

CHAPTER 3
Security Ontologies: A Literature Survey and Classification

Part 2: Main contribution

CHAPTER 4
AMAN-DA: A method for domain specific security requirements engineering

CHAPTER 5
A core security ontology for security requirements engineering

CHAPTER 6
Security goals and security requirements

CHAPTER 7
Domain knowledge and domain ontologies

CHAPTER 8
Ontology based domain security requirements elicitation

Part 3: Evaluation

CHAPTER 9
Case study: Security requirements for the maritime domain (and other domains)

CHAPTER 10
Controlled experiment

LIST OF FIGURES

LIST OF TABLES

Chapitre 1 en français

Introduction

1. SÉCURITÉ : LE CÔTÉ OBSCUR!

Avec la numérisation grandissante des activités dans les différents secteurs (communication, santé, banque, assurance, etc.), les Systèmes d'Information (SI) sont de de plus en plus critiques. Ils doivent se conformer aux usages émergents et à des besoins variés, ils doivent intégrer les nouvelles technologies ce qui les expose en permanence à de nouvelles vulnérabilités. Il n'y a pas une seule semaine sans une annonce indiquant que le SI d'une certaine organisation privée ou publique a été attaqué.
Le coût de la cybercriminalité en 2013 a atteint les 113 milliards de dollars dans le monde (Norton, 2013). Les attaques des SI ciblent des données stratégiques telles que les informations échangées par les directions des systèmes d'information (DSI), les données financières, les documents Recherche et développement (R&D), les informations des clients et des ressources humaines, etc. Les conséquences pour les organisations sont multiples : la détérioration de l'image de marque, la perturbation de l'activité, les pertes financières, ainsi que la menace sur l'éco-système socio-économique, politique et militaire.

La motivation financière de la cybercriminalité est évidente : tandis que le marché annuel des données est estimé à 7 milliards d'euros, les recettes générées par l'exploitation de ces données est d'environ 230 milliards d'euros, avec une croissance annuelle de 40% (Frenkiel, 2009). La plupart des cyber- attaques sont assez connues : dénis de service, virus, chevaux de Troie, hameçonnage, attaques sur les couches réseau inférieures, logiciels espions, etc.

De plus, la presse rapporte régulièrement de mauvaises expériences. Par exemple, en France, en 2010, 150 ordinateurs de la Direction Générale du Trésor ont été attaqués dans le but de capturer des informations relatives à la présidence du G8 et du G20. Fin 2011, le Sénat français a été ciblé par une attaque peu de temps avant l'adoption de la loi sur la reconnaissance du génocide arménien. En juillet 2012, la société DropBox a reconnu que des

mots de passe volés sur un autre site ont permis aux pirates d'accéder aux documents stockés en ligne. En Septembre 2014, cinq millions de mots de passe Gmail et adresses ont été publiées sur un forum (Elyan, 2014). Dans son rapport annuel de 2013 (Symantec, 2013), Symantec signale que la violation de données sensibles a augmenté de 62% avec 253 incidents observés et 552 millions de données d'identité volées. Au cours de l'année 2013, plus de 40 millions de personnes aux États-Unis ont eu leurs informations personnelles volées, ainsi que 54 millions en Turquie, 20 millions en Corée, 16 millions en Allemagne, et plus de 20 millions en Chine.

Il serait simpliste de croire que les vulnérabilités des systèmes d'information sont toujours de nature technologique, ou seulement d'origine interne. En réalité, la sécurité est un problème à multiples facettes ; tout autant que le système d'information visé, la sécurité intègre aussi, dans sa définition, le domaine dans lequel ce système fonctionne.

Tout en développant des techniques de sécurité telles que le cryptage, le contrôle d'identité, ou des architectures spécifiques, notre attention doit être attirée aussi par le contexte sociotechnique dans lequel les systèmes cibles vont fonctionner, dans lequel aussi les menaces et leur danger potentiel peuvent survenir, de manière à découvrir les exigences de sécurité. Dans la pratique, la sécurité est souvent traitée comme une variable d'ajustement (Gruselle, 2013). Comme le montre le rapport de 2014 du (Centre d'analyse stratégique, 2014), les méthodes de gestion des risques sont toujours considérées comme «trop complexes pour être déployées dans les organisations, qui ne sont souvent pas matures en termes de sécurité des SI». Le CLUSIF indique qu'en France, seulement 63% des entreprises ayant plus de 200 employés et moins d'un tiers des collectivités territoriales ont formalisé une politique de sécurité. Même si ces dernières utilisent EBIOS (EBIOS, 2004) et la PSSI (Politique de Sécurité des Systèmes d'Information) de l'ANSSI (Agence Nationale de Sécurité des Systèmes d'Information) quatre fois plus en 2012 qu'en 2008, cela est encore seulement limité à 13% des organisations: 63% d'entre eux n'utilisent même pas une méthode référencée: 43% des entreprises de plus de 200 employés n'utilisent pas une méthode ou ne savent pas quelle méthode utiliser pour formaliser la politique de sécurité. Il est donc nécessaire de proposer des approches simples mais théoriquement et empiriquement fondées qui aident vraiment les organisations à améliorer leurs pratiques et à réduire leur niveau d'exposition aux attaques.

2. PROBLÉMATIQUE

2.1. LA CONNNAISSANCE EST NÉCESSAIRE

Des études récentes (Fenz et Ekelhart, 2009) ont montré que le manque de connaissances liées à la sécurité de l'information au niveau métier est l'une des raisons qui mènent à des stratégies de gestion de sécurité de l'information inadéquates ou inexistantes. De plus, l'effort de sensibilisation à la sécurité et le renforcement des niveaux de connaissance en sécurité conduit à des stratégies plus efficaces. En 2006, l'Agence européenne de la sécurité des réseaux et de l'information (ENISIA) a évalué que l'établissement de bases d'information unifiées pour la gestion des risques de sécurité de l'information, ainsi que la nécessité de méthodes de mesure de risque, étaient des enjeux de grande priorité.

Haley montre que, parmi les principaux défis inhérents aux projets de sécurité, se trouve la difficulté d'expression des exigences de sécurité et de la production de spécifications exhaustives (Haley et al., 2008). En fait, la plupart des développeurs de logiciels ne sont pas intéressés ni informés sur la sécurité (Tondel, Jaatun et Meland, 2008) (Mouratidis, Giorgini, et Manson, 2005), Pendant des décennies, l'accent a été mis sur i) la mise en œuvre des fonctionnalités avant la date limite et ii) l'application de correctifs aux bugs inévitables (Meier, 2006). Cependant, la communauté du génie logiciel et des systeèmes d'information commence à se rendre compte que la sécurité de l'information est également importante pour les logiciels et les systèmes dont la fonction principale n'est pas liée à la sécurité. Donald Firesmith affirme que la plupart des ingénieurs en charge de l'expression des exigences sont insuffisamment formés pour élucider, analyser et spécifier les exigences de sécurité (Firesmith, 2004). Par conséquent, ils confondent souvent des exigences de sécurité avec des mécanismes de sécurité architecturales qui sont traditionnellement utilisés pour satisfaire les exigences. Ils définissent plutôt des choix architecturaux et de conception au lieu de spécifier réellement des exigences.

Zuccato et al. (Zuccato, Daniels et Jampathom, 2011) signalent que l'ingénierie des exigences de sécurité, dans la pratique, est souvent effectuée par des non-spécialistes en sécurité. L'expertise en sécurité est "rare". Les exigences de sécurité et leurs dépendances ne sont souvent pas directement connues par les spécialistes de l'ingénierie des exigences.

2.2. UNE CONNAISSANCE GENERIQUE DE LA SECURITE N'EST PAS SUFFISANTE, ELLE DOIT ETRE SPECIFIQUE AU DOMAINE

Au-delà de la difficulté de maîtriser les exigences de sécurité, il faut prendre conscience du fait que ces dernières sont étroitement liées au domaine d'application. Une bibliothèque publique aura certainement une vision de la sécurité informatique différente de celle d'une chambre de compensation centrale pour les transactions interbancaires. Les exigences de sécurité spécifiques à une installation particulière ne peuvent être déterminées qu'après un examen attentif du contexte de l'entreprise, des préférences de l'utilisateur, et/ou de la posture de défense (Devanbu et Stubblebine, 2000).

La connaissance du domaine, c'est-à-dire la connaissance spécifique au domaine pour lequel les exigences de sécurité sont définies, n'est souvent pas prise en considération, laissant la tâche à l'ingénieur qui définit les exigences ou l'analyste en sécurité d'«imaginer» ou de «capturer» les propriétés du domaine.

Les ingénieurs en charge de la définition des exigences sont souvent considérés comme ayant une connaissance explicite sur la sécurité et sur le domaine pour lequel ils élucident des exigences de sécurité. Or, c'est loin d'être le cas : les domaines évoluent constamment, et leur connaissance est trop complexe à comprendre et à maîtriser.

Certaines recherches précédentes ont pris en considération la connaissance pour l'élucidation des exigences, en particulier avec des ontologies (Daramola et al., 2012a), (Velasco et al., 2009). L'expérience empirique avec RITA (Salinesi, Ivankina et Angole, 2008) a montré que «étant génériques, les menaces dans l'ontologie de RITA ne sont pas spécifiques au domaine cible [banque]» (l'étude de cas était dans le secteur bancaire). Les experts impliqués dans l'évaluation se sont plaints du «manque de spécificité des types de menaces et des exigences de sécurité pour le domaine concerné".

2.3. FONDEES SUR DE LA CONNAISSANCE VERSUS FONDEES SUR DES MODELES

Beaucoup de recherches ont été menées sur les méthodes d'ingénierie des exigences de sécurité fondées sur des modèles (Mouratidis et Giorgini, 2007) (Van Lamsweerde, 2004) (Dahl, Hogganvik et volé, 2007). Les approches fondées sur des modèles peuvent être utiles car elles fournissent une syntaxe formelle pour la spécification des concepts de sécurité. Cependant, nous pensons qu'une approche fondée sur la connaissance peut être une meilleure option car elle permet de représenter, de rendre accessibles et de réutiliser des connaissances

dans le but de développer des méthodes, des techniques et des outils pour l'analyse des exigences de sécurité. Malgré des efforts considérables dans le domaine, à notre connaissance, ces questions ne sont pas encore complètement traitées par les approches existantes. Nous pensons, en particulier, à la plupart des approches fondées sur les modèles (Mouratidis et Giorgini, 2007) (Van Lamsweerde, 2004) (Sindre et Opdahl, 2001a) et aux approches fondées sur l'analyse des risques (Mayer, 2012) (Morali et al., 2009). La même observation peut être faite concernant les approches « industrielles » (EBIOS, 2004) (CLUSIF, 2004) qui utilisent peu la connaissance. Certaines méthodes fondées sur les connaissances existent, notamment celles fondées sur des ontologies (Salinesi, Ivankina et Angole, 2008) (Velasco et al., 2009). Mais elles ne prennent pas en considération le domaine pour lequel les exigences sont définies.

Le problème ciblé dans cette thèse est donc d'exploiter à la fois la connaissance liée à la sécurité et la connaissance spécifique du domaine afin de mieux guider l'élucidation des exigences de sécurité spécifiques à un domaine particulier.

3. CADRE DU PROJET DE RECHERCHE DOCTORAL

La contribution principale de cette thèse doctorale est une méthode pour « définir des **exigences de sécurité** spécifiques à **un domaine** pour un **système d'information** en **réutilisant la connaissance enregistrée** dans des **ontologies** de domaine de sécurité »

Les concepts manipulés au cours de cette recherche appartiennent à différents domaines de recherche, notamment : (a) l'ingénierie des exigences, (b) l'ingénierie des connaissances, (c) la sécurité, (d) l'ingénierie des domaines et des ontologies, (e) l'ingénierie des systèmes d'information. Ces domaines de recherche peuvent fournir des définitions différentes pour des concepts similaires.

Cette section définit chaque concept utilisé au cours de la préparation de cette thèse de doctorat.

3.1. SYSTÈME D'INFORMATION

Il n'y a pas de définition unique des systèmes d'information (Carvalho, 2000). Dans cette thèse doctorale, un système d'information (SI) pour lequel les exigences de sécurité sont élucidées et analysées est défini comme suit : « Un système, soit automatisé ou manuel, qui comprend des personnes, des machines, et/ou des méthodes organisées pour collecter, traiter,

transmettre et diffuser des données (qui représentent les informations de l'utilisateur) dans un environnement donné» (Longeon et Archimbaud, 1999). En adoptant cette définition, nous rejoignons Mayer (Mayer, 2012) lorsqu'il considère que la sécurité des systèmes d'information englobe non seulement la sécurité des systèmes logiciels ou des architectures informatiques, mais aussi celles des personnes et des installations qui jouent un rôle dans le SI, donc dans sa sécurité. L'exemple typique qui caractérise cette déclaration est celui du vol d'un mot de passe d'une base de données grâce à l'ingénierie sociale. Un autre exemple serait la destruction délibérée du serveur de base de données où les informations de l'organisation sont stockées par un attaquant qui déclenche un incendie dans la salle des serveurs.

3.2. SÉCURITÉ

La littérature indique que la sécurité peut avoir deux significations différentes (Firesmith 2007) (Mayer, 2012) : en anglais on distingue security and safety. En français le mot 'sécurité' est utilisé pour désigner les deux sens (certain attribut sureté à safety). Selon l'Université norvégienne de Science et Technologies (NTNU) : la sécurité est la protection contre les incidents intentionnels. Les incidents intentionnels se produisent à cause d'un acte délibéré et planifié, tandis que la 'safety' est la protection contre des incidents aléatoires. Les incidents aléatoires sont des incidents indésirables qui surviennent à la suite d'une ou plusieurs coïncidences (Albrechtsen, 2002). Le dictionnaire d'anglais New Oxford (Pearsall et Hanks, 2001) décrit la 'safety' comme la condition d'être protégé contre ou de ne pas être en capacité de causer un danger, un risque ou un dommage. Il décrit la 'sécurité' comme l'état d'être libre de danger ou de menace. La différence entre la 'sécurité' et la 'safety' n'est pas remarquable ; les deux désignent des conditions dans lesquelles l'on est bien protégé. Pour autant que cette thèse soit concernée, le terme 'sécurité' couvre à la fois les dommages accidentels, ainsi que les dommages délibérés. L'idée de base est de protéger les actifs des risques / menaces et de créer des conditions de sécurité. Ceci est cohérent avec d'autres méthodes de la littérature (décrites dans l'état de l'art de la thèse) qui ne font pas de distinction entre la sécurité et la 'safety' et utilisent une définition générale de la sécurité comme cela se fait ici. La sécurité en tant que discipline couvre un ensemble de méthodes, techniques et outils, permettant la protection des ressources d'un système. Comme défini dans la norme ISO / IEC 13335 (ISO-b, 2004), dans le cadre des systèmes d'information, la sécurité couvre sept critères : la confidentialité, l'intégrité, la disponibilité, l'authenticité, la responsabilité, la non-répudiation et la fiabilité. Le concept de vulnérabilité est considéré

comme une propriété du système ou de son environnement qui, en conjonction avec une attaque, peut conduire à une défaillance de la sécurité (Anderson et Anderson, 2001). Les actifs sont un autre concept dans la littérature de sécurité définis comme quelque chose ayant de la valeur dans une organisation (ISO-b, 2004) qui pourrait être l'objet d'attaques (Schneier, 2003).

Dans cette thèse doctorale, nous sommes intéressés en particulier à la sécurité dans les phases initiales du développement des systèmes, en particulier la phase de définition des exigences de sécurité.

3.3. INGÉNIERIE DES EXIGENCES

Une exigence est définie comme "une condition ou une fonction nécessaire à un utilisateur pour résoudre un problème ou atteindre un objectif" (Pohl, 2010). L'Ingénierie des Exigences (IE) est considérée comme une seule phase du processus de développement de logiciels dont la tâche principale est de développer la spécification des exigences pour le système. L'IE est ainsi réalisée pour chaque projet au début du processus de développement. Les exigences élucidées sont écrites dans un cahier des charges spécifique au projet qui sert alors de document de référence pour les phases suivantes de développement (Pohl, 2010). Firesmith (Firesmith, 2007) définit l'ingénierie des exigences comme : «La discipline de l'ingénierie au sein de l'ingénierie des systèmes/logiciels concernés par l'identification, l'analyse, la réutilisation, la spécification, la gestion, la vérification et la validation des buts et des exigences (y compris les exigences liées à la sécurité)".

Dans cette thèse doctorale, toutes ces définitions sont acceptées ; les exigences que nous avons l'intention de définir sont des conditions énoncées par des parties prenantes pour atteindre leurs buts. En outre, nous avons l'intention de développer des spécifications (textuelles et de modèles) qui serviront de références pour les phases ultérieures de développement. Cependant, dans notre travail de recherche, nous nous concentrons sur l'élucidation, l'analyse et la réutilisation des exigences. Nous ne nous attaquons pas à la vérification ou validation telle que décrite par Firesmith. Le travail de thèse se concentre exclusivement sur l'élucidation, l'analyse et la réutilisation des exigences.

3.4. L'INGENIERIE DES EXIGENCES DE SECURITE POUR UNE MEILLEURE GESTION DE LA SECURITE

Au cours de la dernière décennie, la communauté scientifique a commencé à appeler à un examen précoce de la sécurité, au cours de la phase d'ingénierie des exigences. Considérant la sécurité durant les premiers stades du développement des SI permet aux développeurs d'envisager les menaces, leurs conséquences et les contre-mesures avant qu'un système soit en place, plutôt qu'une éventuelle réaction lorsque une attaque désastreuse se produit (Mayer, 2012). Cette attention a donné naissance à un domaine croissant de recherche appelé l'Ingénierie des Exigences de Sécurité (IES).

(Haley et al., 2008) définit les exigences de sécurité selon trois catégories : 1) comme des fonctions, 2) comme des exigences non-fonctionnelles, 3) selon d'autres interprétations. Les trois sections suivantes abordent la littérature conformément à cette catégorisation :

A. Les exigences de sécurité en tant que fonctions

Les exigences de sécurité sont parfois exprimées sous la forme de mécanismes de sécurité à utiliser. Par exemple, la version ISO des Critères Communs (ISO-a, 1999) fournit des exemples de formulation générale, comme par exemple :

> "La [...] Fonction de Sécurité (LFS) doit refuser explicitement l'accès de sujets à des objets en fonction de la [règles ...] "

Où [règles] fait réféence à un mécanisme. Le NIST Handbook Computer Security (Guttman et Aoback, 1995) stipule que «Ces exigences [de sécurité] peuvent être exprimées comme des caractéristiques techniques (par exemple, les contrôles d'accès), des assurances (par exemple, la vérification des antécédents pour les développeurs de systèmes), ou des pratiques opérationnelles (par exemple, sensibilisation et formation) ». En d'autres termes, les exigences sont spécifiées en termes de fonctions et de procédures.

La définition des exigences en termes de fonctions laisse de côté des informations clés : quels objets doivent être protégés et, plus important encore, pourquoi les objets doivent être protégés. Les deux documents ISO et NIST apportent peu d'indications sur la manière de connecter les fonctionnalités (coté solution) avec les exigences de sécurité (coté problème). Au lieu de décrire quand et pourquoi les objets doivent être protégés, ils décrivent comment les objets doivent être protégés, ce qui est en désaccord avec la démarche de l'IE qui recommande de s'intéresser à l'espace du problème avant de plonger dans l'espace de solution.

B. Les exigences de sécurité comme des exigences non fonctionnelles (ENFs)

Devanbu et Stubblebine (Devanbu et Stubblebine, 2000) remarquent que les exigences de sécurité sont un type d'exigence non-fonctionnelle. Kotonya et Sommerville (Kotonya et Sommerville, 1998), lors de l'examen des exigences non-fonctionnelles, dans lesquelles sont incluses celles de sécurité, les définissent comme «des restrictions ou contraintes» sur les services du système. Rushby (Rushby, 2001) stipule que : «Les exigences de sécurité concernent essentiellement ce qui ne doit pas arriver. » En utilisant la méthode Tropos, Mouratidis et al (Mouratidis, Giorgini, et Manson, 2003a) mentionnent que : «les contraintes de sécurité définissent les exigences de sécurité du système".

Le problème engendré par ces définitions est leur manque de spécificité : En IE, il ne suffit pas de dire ce que l'on ne veut pas. Les buts et leur mise en contexte doivent être explicites aussi.

En d'autres termes, les exigences de sécurité devraient aussi indiquer comment les «services système» sont contraints, l'effet de la contrainte sur la fonctionnalité du système ; et comment on peut valider le système contre toute contrainte potentielle pour assurer qu'il reflète fidèlement la volonté des parties prenantes.

C. D'autres interprétations des exigences de sécurité

Plusieurs auteurs semblent supposer que les exigences de sécurité sont identiques aux buts de sécurité de haut niveau. Tettero et al. (Tettero et al., 1997) définissent les exigences de sécurité en termes de confidentialité, d'intégrité et de disponibilité de l'entité pour laquelle une protection est nécessaire. Bien que ce soit une définition claire, dans certains cas, elle peut ne pas aboutir à des exigences assez précises.

Tout comme dans l'IE en général, les buts sont utiles pour élucider des exigences, les analyser, les justifier, et les tracer. Toutefois, les exigences sont des buts seulement lorsqu'il est mentionné si 'elles doivent être opérationnalisées sur le système et comment (Van Lamsweerde, 2009).

Un autre terme utilisé pour les exigences de sécurité est "politique de sécurité". Par exemple, Devanbu et Stubblebine (Devanbu et Stubblebine, 2000) définissent une exigence de sécurité comme "une traduction d'une politique organisationnelle de haut niveau en exigences détaillées d'un système spécifique. (Anderson et Anderson, 2001) affirment que la politique de sécurité est "un document qui exprime [...] ce que [...] les mécanismes de protection ont à

atteindre» et que «le processus d'élaboration d'une politique de sécurité [...] est un processus de l'ingénierie des exigences ».

Dans cette thèse doctorale, nous faisons la distinction entre *critères de sécurité*, *buts de sécurité* et *exigences de sécurité*.

Les critères de sécurité définissent des propriétés de sécurité telles que la confidentialité, l'intégrité, la disponibilité et la traçabilité. Un *but de sécurité* définit ce que les parties prenantes espèrent atteindre à l'avenir en termes de sécurité. Ils sont à un haut niveau d'abstraction, mais ne sont pas toujours bien formalisées. Par exemple, une partie prenante pourrait exprimer le but de maintenir la confidentialité des renseignements personnels.

Les exigences de sécurité sont des conditions définies sur l'environnement qui doivent être respectées afin d'atteindre un objectif de sécurité (ils matérialisent un but de sécurité) et permettent de diminuer le risque.

En cohérence avec la définition de Glinz des ENF (Exigences Non Fonctionnelles) (Glinz, 2007) les exigences de sécurité peuvent être des exigences non-fonctionnelles à un certain niveau d'abstraction et fonctionnel à un autre niveau d'abstraction. Cela dépend de ce que nous voulons protéger et du niveau de sécurité cible. Les exigences de sécurité peuvent être liées aux bases de données, aux applications, aux systèmes d'exploitation, aux organisations et à l'environnement extérieur. Cette thèse utilise aussi par fois le terme «exigence de sécurité», pour englober à la fois «politique de sécurité» et «exigence de sécurité».

3.5. QU'EST-CE QUE LA CONNAISSANCE ?

La connaissance peut être définie comme une croyance justifiée qui augmente la capacité d'une entité pour une action efficace (TM Kusuma, 2013). Selon Debenham, "La connaissance représente les associations fonctionnelles explicites entre les éléments d'information et/ou des données" (Debenham, 1989).

La connaissance doit être différenciée de la donnée et de l'information. Les données (data est le pluriel de *datum*) sont des faits bruts (Long et Long, 1998). Par exemple ; "Signaux bruts" sont les données. L'information est le sens que nous donnons à des faits accumulés (données) (Long et Long, 1998). "Alerte SOS " est le sens attaché aux «signaux bruts». C'est une information.

La connaissance attache de la compétence et un but à l'information, et il est possible d'en générer une action. Ici : alerte d'urgence SOS → commencer l'opération de sauvetage (une alerte d'urgence génère l'action de démarrage d'une opération de sauvetage).

La connaissance peut être perçue à partir de plusieurs points de vue (TM Kusuma, 2013) :

- Un état d'esprit - la connaissance est l'état de connaitre et de comprendre.
- Un objet - la connaissance est un objet qui peut être stocké et manipulé.
- Un processus - la connaissance est un processus d'application de l'expertise.
- Une condition - la connaissance est l'accès au contenu et son extraction.
- Une capacité - la connaissance désigne le potentiel d'influencer l'action.

La connaissance tacite est profondément enracinée dans les actions, l'expérience et l'implication dans un contexte spécifique. Elle consiste en éléments cognitifs (modèles mentaux) et éléments techniques (savoir-faire et compétences applicables au travail spécifique).
Les connaissances explicites se réfèrent à la connaissance qui est transmissible dans un langage formel et systématique.

L'ingénierie de connaissance est la discipline qui consiste à intégrer les connaissances dans les systèmes informatiques afin de résoudre des problèmes complexes exigeant normalement un niveau élevé d'expertise humaine (Feigenbaum et Pamela 1983).
L'ingénierie de connaissance implique normalement quatre étapes distinctes dans le transfert des connaissances humaines au sein de systèmes à base de connaissances :

(i) la validation des connaissances,
(ii) la représentation des connaissances,
(iii) la représentation d'inférences,
(iv) l'explication et la justification.

Un ingénieur de la connaissance est responsable d'obtenir des connaissances des experts humains, puis d'entrer cette connaissance dans un systèms à base de connaissances.
C'est en 1982 que Newell a publié un article sur "Le niveau de connaissances" (Newell, 1982) dans lequel il a fait valoir la nécessité d'une description de la connaissance à un niveau supérieur à celui des symboles dans les systèmes de représentation des connaissances. Le niveau de connaissance était sa proposition pour réaliser une description d'un système d'intelligence artificielle en fonction de son comportement rationnel : Pourquoi le système

(l'«agent») effectue cette "action", indépendamment de sa représentation symbolique dans les règles, cadres ou logiques (le niveau "symbole").

Dans les années quatre-vingt-dix, les efforts en termes d''ingénierie des connaissances se sont progressivement déplacés vers la connaissance de domaine, en particulier les représentations réutilisables sous la forme d'ontologies. Un article clé, qui a reçu une assez large attention même en dehors de la communauté de l'ingénierie des connaissances est l'article de Gruber sur les ontologies portables (Gruber, 1995). Une ontologie, étant une «représentation d'une conceptualisation" (Gruber, 1995), est une représentation formelle des entités et des relations qui existent dans certains domaines. Il représente une conceptualisation partagée afin de répondre à toutes fins utiles (Dobson et Sawyer, 2006). L'analyse ontologique clarifie la structure de la connaissance. Pour un domaine donné, son ontologie constitue le cœur de tout système de représentation des connaissances de ce domaine. Sans ontologies, et sans conceptualisations qui soient la base de la connaissance, il ne peut y avoir un vocabulaire de représentation et de la réutilisation des connaissances (Chandrasekaran et al, 1999).

Dans cette thèse, nous nous intéressons à deux types de connaissances, les connaissances relatives à la sécurité nécessaires pour effectuer une élucidation et une analyse des exigences de sécurité, et les connaissances du domaine, à savoir le domaine pour lequel les exigences de sécurité sont élucidées (par exemple, banque, santé, maritime, etc.). Dans le cadre de nos travaux de recherche, nous nous sommes particulièrement intéressés à des ontologies comme forme principale de représentation et de réutilisation des connaissances.

3.6. La connaissance du domaine et de la sécurité en ingénierie des exigences

A. Qu'est-ce que 'le domaine'?

(Bjørner, 2010b) définit un domaine comme une «zone» d'activité humaine, naturelle, ou les deux, où la «zone» est «bien délimitée», comme, par exemple, pour la physique: la mécanique ou l'électricité ou la chimie ou de l'hydrodynamique; pour un composant de l'infrastructure: les banques, les chemins de fer, l'hôpital; ou pour un marché: les consommateurs, les détaillants, les grossistes, les producteurs et la chaîne logistique.

Cette thèse rejoint Bjørner en considérant le domaine comme un univers de discours, petit ou grand, à savoir une structure (i) d'entités, de choses, d'individus, de renseignements dont certains peuvent avoir différents états; (ii) de fonctions, sur les entités qui, appliquées deviennent éventuellement des actions de changement d'état du domaine; (iii) d'événements, impliquant peut-être des entités, se produisant dans le temps et exprimables comme des

prédicats uniques ou paires de (avant/après) états; et (iv) de comportements, d'ensembles de séquences éventuellement interdépendantes d'actions et d'événements.

B. Pourquoi la connaissance du domaine est importante pour l'ingénierie des exigences ?

Est-il possible d'élaborer des exigences du système sans comprendre le domaine ? Alors que certains auteurs dans la littérature ont rapporté que cela est impossible (Kaiya et Saeki, 2006) (Bjørner, 2010b), d'autres affirment que l'ignorance est importante aussi (Berry, 1995). Cette thèse commence par l'observation que les analystes, les consultants, « designers », architectes, développeurs mettent en place des systèmes pour les hôpitaux (resp. chemins de fer, banques) avec peu de connaissance liées à la santé (resp. des transports, des marchés financiers). Cela n'est pas le cas dans toutes les autres disciplines de l'ingénierie où la connaissance professionnelle est très enracinée : un ingénieur aéronautique comprend le domaine de l'aérodynamique ; les architectes navals (c.-à-d, concepteurs de navires) comprennent le domaine de l'hydrodynamique ; les ingénieurs des télécommunications comprennent le domaine de la théorie des champs électromagnétiques ; et ainsi de suite.

Selon (Kaiya et Saeki, 2006), la connaissance du domaine détenue par des experts de ce domaine joue un rôle important dans l'élucidation des exigences de haute qualité. Bien que les analystes en exigences ont beaucoup de connaissances sur la technologie des logiciels, leurs connaissances sur le domaine pour lequel le logiciel est à développer sont souvent tacites, pas clairement définies, ni formalisées. Le manque de connaissance du domaine peut entraîner une pauvre élucidation des exigences.

Pour toutes ces raisons, nous suivons Bjørner dans ses déclarations : «avant que le logiciel soit conçu, nous devons comprendre les exigences. Avant que les exigences puissent être finalisées, nous devons avoir compris le domaine». Comme le montre la figure F.1.1, ceci impacte le cycle de vie du développement des systèmes puisque l'ingénierie de domaine précède l'ingénierie des exigences. Les résultats de l'ingénierie de domaine comprennent un modèle de domaine : une description qui peut être à la fois informelle, comme un récit en langue naturelle et formelle, comme des spécifications. L'ingénierie des exigences aboutit, entre autres, à un modèle d'exigences. Les résultats de la conception de logiciels incluent un code exécutable et toute la documentation qui l'accompagne.

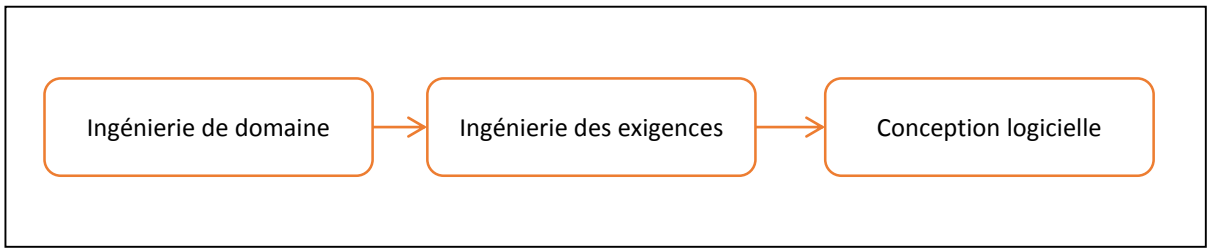


Figure F.1.1. Le triptyque du développement logiciel selon Bjorner.

C. Pourquoi la connaissance du domaine est-elle importante pour l'ingénierie des exigences de sécurité ?

Les exigences de sécurité et l'ingénierie des exigences de sécurité n'échappent pas à l'observation faite ci-dessus. L'élucidation des exigences de sécurité des parties prenantes est en réalité l'une des étapes les plus difficiles dans le processus d'analyse des exigences, et plusieurs méthodes et outils informatiques ont été étudiés et développés afin de soutenir les activités humaines de l'élucidation des exigences de sécurité. Cependant, ces méthodes et outils sont trop généraux ou bien la connaissance spécifique au domaine n'est pas exploitée. Puisque la sécurité n'est pas une question à part, au moment de définir les exigences de sécurité, l'ingénieur a besoin de savoir quel est le périmètre organisationnel de l'étude, quels sont les actifs qui ont besoin de protection et quelles sont les différentes personnes impliquées dans l'organisation. Toutes ces informations relèvent du domaine concerné. De plus, les contre-mesures pour remédier aux risques diffèrent selon que l'organisation appartient au domaine de la santé, de la banque, de l'aéronautique, etc. Connaitre les contre-mesures spécifiques à chaque domaine aide à élucider des exigences de sécurité de meilleure qualité.

3.7. La réutilisation de la connaissance en ingénierie des exigences

En 1993, le deuxième atelier international sur la réutilisation logicielle s'est tenu à Lucca, en Italie. La plupart des recherches présentées lors de cet événement étaient axées sur la réutilisation de code, de design ou d'architecture. En d'autres termes, l'idée était que principalement les artefacts « durs » – code, objet et ainsi de suite- pourraient être réutilisés. Très peu d'auteurs ont examiné l'idée de la réutilisation plus tôt dans le cycle de vie, à savoir la réutilisation des exigences elles-mêmes. Dans les vingt dernières années en revanche, sont apparus des bibliothèques de réutilisation, des méthodes d'ingénierie de domaine et des outils : pour le design de réutilisation, pour des architectures de logiciel spécifique, pour des

modèles de conception de domaine, pour des générateurs, pour de la mesure et de l'expérimentation (Frakes et Kang, 2005).

De nos jours, la pratique de la réutilisation se déplace en amont et la réutilisation concerne aussi les artefacts les plus abstraits. Les exigences sont souvent recyclées ; les modèles sont échangés sur Internet. La notion de réutilisation à l'étape des exigences est largement acceptée par beaucoup au sein de la communauté comme un objectif souhaitable (Lam, McDermid et Vickers, 1997). Par exemple, une conférence de travail sur les patrons (Pattern Languages of Programs) se tient deux fois par an et les résultats sont le partage des connaissances et la publication de nouveaux patrons (Robertson et Robertson, 2013)

Contribuer à la réutilisation des exigences peut consister soit en la récupération d'exigences qui ont été écrites pour des projets précédents, puis en leur adaptation pour un nouveau projet, soit en l'écriture d'exigences à partir de zéro à un niveau raisonnable de généralité et d'abstraction afin de pouvoir les utiliser dans différents projets. Une distinction claire doit être faite entre l'ingénierie « pour » la réutilisation et l'ingénierie « par » la réutilisation (Maiden, 1993).

Par exemple, il est possible de réutiliser différents types de données, allant des exigences métier et des exigences fonctionnelles à l'utilisation des cas, notamment des cas de test. Comme l'ingénierie des exigences est la première phase dans le processus de développement de logiciels, la réutilisation des exigences peut renforcer le cycle de vie du logiciel. Des recherches antérieures (López, Laguna et Peñalvo, 2002) ont souligné que la réutilisation des premiers produits et processus implémentés dans un projet de développement logiciel peut avoir un impact sur le cycle de vie de deux facons : (a) en permettant aux ressources du développement logiciel d'être plus profitables, et (b) en promouvant un développement basé sur la réutilisation dans le processus logiciel.

De plus, à un haut niveau d'abstraction, chaque application a tendance à avoir les mêmes types de base d'actifs vulnérables (par exemple, les données, les communications, les services, les composants matériels et personnel). De même, ces actifs vulnérables ont tendance à être soumis aux mêmes types de base de menaces à la sécurité (par exemple, le vol, le vandalisme, la divulgation non autorisée, la destruction, la fraude, l'extorsion, l'espionnage, etc.) par des attaques des mêmes types de base d'attaquants (par exemple, hackers, crackers, employés mécontents, cyber-terroristes, espions industriels, etc.) qui peuvent être profilées avec des motivations et des niveaux typiques d'expertise et d'outils. Les exigences de sécurité ont tendance à être encore plus normalisées que leurs mécanismes

associés. Par exemple, pour répondre aux exigences d'identification et d'authentification, on peut avoir plusieurs choix de mécanismes d'architecture au-delà de l'identifiant et du mot de passe (Firesmith, 2004). Sur cette base, les exigences de réutilisation pourraient conduire à d'importantes économies de temps et de coûts de développement (Hermoye, van Lamsweerde et Perry, 2006). Structurer les connaissances de sécurité aide l'utilisateur de ces connaissances à en parcourir le contenu et à trouver les informations pertinentes de manière plus efficace. Différentes représentations des connaissances existent dans la littérature. Des « patterns » d'attaques récurrentes et vulnérabilités ont été identifiés par les praticiens de logiciels de sécurité (Mead et Stehney, 2005). Des modèles de sécurité d'un haut niveau d'abstraction ont également été introduits à des fins de réutilisation (Firesmith, 2004). Diverses autres approches pour la gestion des connaissances et leur réutilisation dans la sécurité existent dans la littérature, telles que des taxonomies, des ontologies, des normes et des lignes directrices. Au cours de la dernière décennie, les chercheurs ont accordé une certaine attention aux avantages de la réutilisation de la connaissance dans le processus de l'IES étant donné la nature commune des problèmes de sécurité dans les applications et dans les domaines (Firesmith, 2004). La connaissance de la sécurité est difficile à acquérir. En plus de la sensibilisation sur les attaques potentielles, la conception de systèmes critiques pour la sécurité nécessite des connaissances et de l'expertise en matière de sécurité dans diverses techniques telles que les réseaux informatiques, les systèmes d'exploitation, les protocoles de communication, les algorithmes de cryptographie et les méthodes de contrôle d'accès. La réutilisation combinée à une connaissance structurée prédéfinie peut rendre le travail des ingénieurs, en charge de la spécification des exigences, beaucoup plus facile et plus rapide, car ils manquent généralement d'expertise et de compétences en sécurité. Cependant, il faut être prudent lors de la structuration des connaissances réutilisables - le résultat doit être d'une grande qualité. Sinon, cela pourrait générer de nouveaux problèmes de sécurité.

Dans cette thèse doctorale, la connaissance en sécurité inclut des menaces et des attaques, des vulnérabilités, des exigences de sécurité. La connaissance du domaine inclut des entités comme des acteurs, des organisations, des actifs, des emplacements, etc.

4. QUESTIONS DE RECHERCHE

La thèse aborde les problèmes mentionnés ci-dessus en proposant une méthode qui guide les ingénieurs (ainsi que les analystes de sécurité) dans l'élucidation des exigences de sécurité

pour des domaines spécifiques. La méthode est générique dans le sens oùelle peut être appliquée à divers domaines, mais, lors de son application, elle devient spécifique à un domaine.
Ainsi, l'objectif principal de cette thèse est de répondre aux questions de recherche suivantes :

Principale question de recherche : *Comment élucider des exigences de sécurité qui sont spécifiques à un domaine avec une stratégie de réutilisation ?*

Pour répondre cette question de recherche principale, quatre sous-questions doivent être considérées :

RQ1. Comment structurer la connaissance sur la sécurité à la fois au niveau générique et en relation avec les domaines spécifiques ?
RQ2. Comment structurer la connaissance relative aux domaines à la lumière des préoccupations de sécurité ?
RQ3. Quelle est la meilleure façon de capturer les buts et exigences des parties prenantes en termes de sécurité ?
RQ4. Comment exploiter les deux connaissances, celle liée à la sécurité et celle liée au domaine, pour produire des exigences spécifiques aux domaines et construire des modèles d'exigences ?

5. HYPOTHÈSES DE RECHERCHE

Le but principal des travaux de recherche décrits dans cette thèse est l'élucidation des exigences de sécurité spécifiques aux domaines grâce à une stratégie de réutilisation. Différentes approches peuvent être adoptées pour atteindre cet objectif. Une approche possible pourrait consister à dériver, à partir de la connaissance de sécurité générique, une connaissance de la sécurité spécifique au domaine, et ensuite utiliser cette connaissance de sécurité spécifique au domaine pour l'élucidation des exigences de sécurité. Une autre approche pourrait être d'utiliser les connaissances de la sécurité et du domaine pour obtenir des exigences de sécurité spécifiques au domaine. Dans cette thèse, la seconde approche est développée. Toutefois, avec cette approche, des questions se posent concernant la meilleure

structure de la connaissance, et la façon d'utiliser ces connaissances pour construire des modèles d'exigences de sécurité.

Pour résumer, le travail de cette thèse teste les principales hypothèses suivantes :

a. Il est possible de fournir la connaissance de sécurité nécessaire en choisissant des structures adéquates.
b. Il est possible de construire des modèles d'exigences de sécurité en s'appuyant sur la connaissance de sécurité et de domaine.
c. S'appuyer sur la connaissance du domaine en plus de la connaissance de sécurité va apporter une valeur ajoutée dans la production d'exigences de sécurité spécifiques au domaine en s'appuyant sur des connaissances de sécurité génériques.
d. L' approche décrite en 3 est efficace.
e. Une bonne implémentation de cette approche est possible et utile pour les utilisateurs finaux.

6. MÉTHODE DE RECHERCHE

Le travail de recherche décrit dans cette thèse a été réalisé en utilisant une méthode de science de la conception (design science). (Hevner et Chatterjee, 2010) définissent la « design science » comme étant la conception et la validation de propositions de solutions à des problèmes pratiques. Hevner et al. (Hevner et Chatterjee, 2010) suggèrent que la « design science » diffère sur deux aspects d'autres branches de la science : (a) elle s'intéresse à des objets (artéfacts) plutôt qu'à des faits de structure de la nature ou sociale, et (b) elle cible la recherche de règles prescriptives pour la conception, plutôt que la recherche de descriptions, d'explications et de prévisions, comme c'est le cas dans les autres branches de la science. Etant orientée vers la conception, la méthode de recherche utilisée dans cette thèse consiste à valider les hypothèses de recherche présentés ci-dessus au moyen d'études de cas et d'expériences contrôlées.

Pour répondre à la question de recherche principale et pour tester les hypothèses de recherche de cette thèse, nous avons mené une stratégie de recherche basée sur le modèle de processus de « design science » proposé par (Peffers et al., 2007). Ce processus comporte six étapes principales : identification et motivation du problème, définition desobjectifs d'une solution, conception et développement, démonstration, évaluation, communication). La figure F.1.2

présente le modèle de processus de « design science » pour la recherche en système d'information, et l'application de ce processus à la recherche menée dans cette thèse.

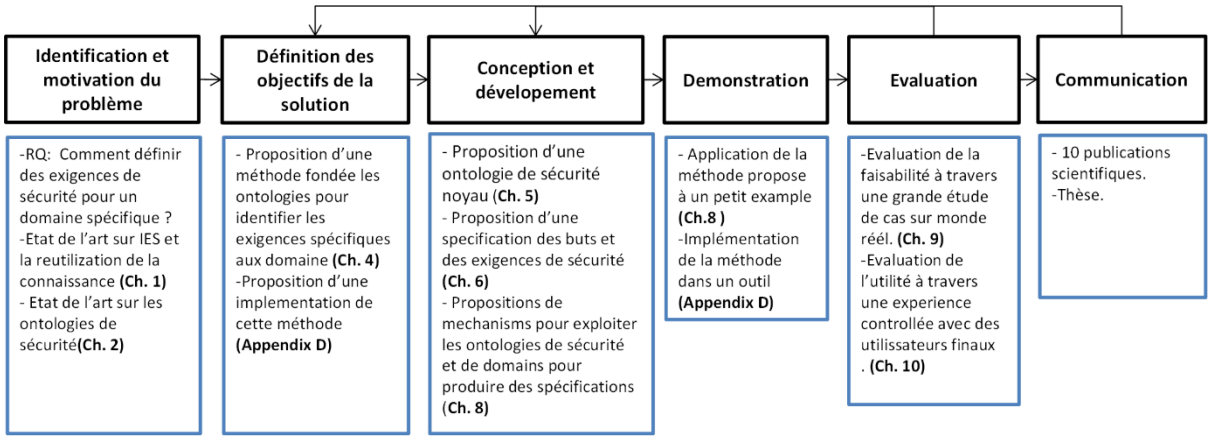


Figure F.1.2. Application du modèle de processus de design science pour la recherche en système d'information (Peffers et al. 2007) à cette thèse.

Plus précisément, la stratégie suivante a été mise en œuvre :

A. Une étude de cartographie systématique sur la réutilisation des connaissances dans l'ingénierie des exigences de sécurité a été menée pour (i) mettre à jour les synthèses de la littérature liées à l'IES avec les recherches plus récentes, (ii) identifier la connaissance (ré) utilisée dans l'IES, (iii) distinguer les différents types de structures permettant la réutilisation des connaissances dans l'IES, (iv) comprendre leur utilisation, et (v) faire une synthèse des définitions des exigences de sécurité spécifiques aux domaines par les méthodes de l'IES.
B. Une revue de littérature sur les ontologies de sécurité a été effectuée. L'objectif principal était d'examiner, d'analyser, de sélectionner et de classer les ontologies de sécurité, comme une étude de portée générale, mais avec un intérêt particulier dans le domaine de l'ingénierie des exigences de sécurité.
C. Une synthèse des lacunes et des inconvénients présents dans les approches existantes a été effectuée.
D. Une méthode fondée sur les ontologies a été proposée pour analyser et élucider des exigences de sécurité spécifiques aux domaines. Dans la méthode proposée, trois sous-propositions ont été développées: l'ontologie de sécurité noyau, la spécification des buts de sécurité et des exigences de sécurité, et une technique à base de règles pour explorer l'ontologie de domaine et de sécurité au cours du processus délucidation des exigences.

E. La faisabilité de la méthode proposée a été évaluée par une étude de cas réelle. La facilité d'utilisation et l'utilité de la méthode proposée ont été évaluées avec des utilisateurs finaux. Les résultats de ces évaluations permettent d'appuyer ou de réfuter les hypothèses proposées dans cette thèse.
F. De nouvelles directions de recherche ont été identifiées. Les résultats de l'étude de cas et de l'expérience contrôlée ainsi que le retour de la communauté des systèmes d'information ont été prises en compte pour améliorer la méthode initiale.

7. CONTRIBUTIONS

Cette thèse produit quelques contributions originales :

- Un cadre d'étude pour analyser et comparer la réutilisation des connaissances dans l'IES. Le cadre d'étude a été proposé après une étude cartographique systématique de la réutilisation des connaissances dans l'ingénierie des exigences de sécurité. L'étude cartographique systématique a analysé plus de 150 publications au cours des vingt dernières années. Les différentes formes de réutilisation dans l'ingénierie des exigences de sécurité (ontologies, taxonomies, patterns, modèles génériques, ..) ont été identifiées, les techniques de réutilisation, ainsi que la connaissance réutilisée. Ce travail a été publié dans la revue internationale d'ingénierie des exigences (Requirements Engineering Journal).
- Un cadre d'étude pour classer les ontologies de sécurité. Ainsi, huit familles d'ontologies de sécurité ont été identifiées après une lecture profonde d'environ 50 publications. Les huit familles sont les suivantes : ontologies initiales de sécurité, taxonomies de sécurité, ontologies générales de sécurité, ontologies de sécurité spécifiques, orientées Web, basées sur le risque, de modélisation, et pour les exigences. Chacune des ontologies de sécurité de l'étude a été analysée du point de vue de sa couverture des aspects de sécurité. Cette contribution a été présentée lors du troisième atelier international sur l'ingénierie de sécurité des systèmes d'information (WISSE'12).
- Une ontologie noyau et une méthode à base de règles pour l'élucidation des exigences de sécurité spécifiques aux domaines. La méthode prend en entrée l'ontologie de sécurité noyau et toute ontologie de domaine pour produire des exigences de sécurité spécifiques textuelles ainsi que des modèles de sécurité. Cette méthode se base sur quatre sous-contributions principales:

- Une ontologie de sécurité noyau. Il s'agit d'une nouvelle méta-ontologie pour l'ingénierie des exigences de sécurité. Elle combine des éléments d'ontologies existantes pour créer une ontologie de sécurité de base. Une description de l'ontologie de sécurité noyau est proposée ainsi que le processus de construction. Plusieurs éléments de l'ontologie sont définis et leurs relations sont présentées. Une évaluation est présentée dans le but de prouver l'exhaustivité, la validité et l'utilisabilité de l'ontologie. Cette contribution a été présentée au Symposium international sur l'ingénierie de logiciels et systèmes sûre en Mars 2015.
- Une ontologie multi-niveau de domaine pour structurer la connaissance du domaine. Cette structure multi-niveaux permet d'éviter à la méthode d'être restreinte à un seul domaine. L'avantage est d'offrir la possibilité de passer d'un domaine à un autre avec la même méthode.
- Des « templates » linguistiques pour spécifier les buts de sécurité et les exigences de sécurité. Ici, les différences entre les buts et les exigences de sécurité sont clarifiées.
- Un mécanisme fondé sur des règles pour extraire des connaissances pertinentes à partir des ontologies de domaine et de sécurité et produire des modèles d'exigences de sécurité et des spécifications textuelles.

La méthode a été évaluée. Sa faisabilité a été évaluée au moyen d'une étude de cas réelle. Son utilité et son utilisabilité ont été évaluées par le biais d'une expérience contrôlée. Différentes parties de la méthode ont été publiées et présentées dans divers événements et éditions. Une des publications a obtenu le prix du meilleur article. Ci-dessous la liste des publications :

- Vers une nouvelle génération de définition des Exigences de sécurité fondée sur l'utilisation des ontologies. **Prix du meilleur article du séminaire doctoral** (Informatique des Organisations et des Systèmes d'Information et de Décision - INFORSID 2012, Montpellier, France).

- Une méthode de définition des Exigences de sécurité Fondée sur l'utilisation des ontologies. (Séminaire de doctorat du Forum Académie -.Industrie de l'AFIS, 2012, Paris, France)

- Towards a new generation of security requirements definition methodology using ontologies (Conférence sur l'ingénierie des systèmes d'information avancés, CAiSE 2012 consortium doctoral, Gdansk, Pologne)

- Using Security and Domain ontologies for Security Requirements Analysis. (Computer Software and Applications Conference Workshops (Atelier de la conférence sur les applications logiciel et informatiques - COMPSACW 2013, Kyoto, Japon).

- Security requirements analysis based on security and domain ontologies. (Ingénierie des exigences - Fondation pour la qualité logicielle - REFSQ 2013, Essen, Allemagne).

- A Methodology for Defining Security Requirements using Security and Domain Ontologies (Insight Journal, INCOSE - Conseil international sur l'ingénierie des systèmes, 2013).

La liste complète des publications produites par cette thèse peut être consultée à l'annexe G.

8. L'ORGANISATION DE LA THÈSE

Cette thèse est organisée en trois parties et dix chapitres comme suit :

Partie 1 : Etat de l'art

La première partie présente l'état de l'art et contient 2 chapitres :

Le Chapitre 2 recense les travaux liés présentés dans la littérature concernant l'ingénierie des exigences de sécurité et la réutilisation des connaissances. Ce chapitre définit sept questions de recherche concernant l'état de l'art sur la réutilisation des connaissances dans l'IES, sa forme de représentation, la connaissance réutilisable, la technique de réutilisation de la connaissance, la dépendance à la connaissance de domaine, les lacunes et les défis trouvés dans la littérature. Ces questions ont été traitées au moyen d'une étude cartographique systématique présentée dans le chapitre.

Le Chapitre 3 synthétise les travaux présentés dans la littérature liés aux ontologies de sécurité. Le chapitre soulève trois questions principales sur les différentes ontologies de sécurité existantes de nos jours, sur leur couverture des aspects de sécurité, sur 'la meilleure' ontologie de sécurité qui peut être choisie pour la définition des exigences SI.

Partie 2 : La contribution principale

La deuxième partie de cette thèse présente le corps principal de la contribution. Elle contient cinq chapitres principaux :

Le Chapitre 4 fournit un aperçu sur la méthode d'élucidation des exigences de sécurité spécifiques au domaine. Elle présente brièvement les principaux éléments sur lesquels la méthode se base. Le chapitre introduit un exemple motivant qui va être utilisé dans le reste de la thèse pour illustrer la méthode.

Le Chapitre 5 présente la première contribution de cette thèse doctorale : une ontologie de sécurité noyau pour l'élucidation des exigences de sécurité. L'ontologie est présentée, ses concepts et relations sont décrits. La complétude de l'ontologie est validée par rapport aux autres ontologies. L'ontologie est validée à l'aide de quelques requêtes indicatives.

Le Chapitre 6 décrit les buts et les exigences de sécurité. Elle présente le modèle de but de sécurité et le pattern syntaxique proposés pour manipuler les exigences de sécurité dans la méthode.

Le Chapitre 7 décrit l'ontologie de domaine multi-niveaux proposée pour englober la connaissance du domaine. L'ontologie de haut niveau (upper) permet de passer d'un domaine un autre.

Le Chapitre 8 présente comment les artéfacts précédents (ontologie de sécurité, buts et exigences de sécurité, ontologie de domaine multi-niveaux) ont été mis en place dans la méthode pour produire des buts de sécurité et des spécifications textuelles. Le chapitre présente un algorithme et les règles nécessaires (règles de cartographie et de production) pour extraire les connaissances pertinentes des ontologies. Dans le chapitre, le processus est présenté étape par étape à l'aide d'un exemple d'exécution.

Partie 3 : Évaluation

La troisième partie de la thèse de doctorat décrit l'évaluation de la proposition. Il contient deux chapitres :

Le Chapitre 9 présente l'évaluation de la faisabilité et la généricité de la méthode. Il décrit l'application de la méthode à une étude de cas réelle liée au domaine maritime. Le chapitre examine la généricité de la méthode à travers deux autres applications.

Le Chapitre 10 présente l'évaluation de la facilité d'utilisation et de l'utilité de la méthode. Cela se fait grâce à une expérience contrôlée avec des utilisateurs finaux. Le chapitre présente les résultats obtenus et discute les retours recueillis.

Chapter 1

Introduction

1. SECURITY: THE DARK SIDE!

With ever-growing digitization of activities in various sectors (communication, health, banking, insurance, etc.), Information Systems (IS) are getting more and more critical. They must comply with emerging usages and varied needs and integrate new technologies, while, at the same time, they are permanently exposed to new vulnerabilities. Not a single week goes by without an announcement indicating that the IS of some private or public organization was attacked.

The cost of cybercrime in 2013 is up to 113B$ worldwide (Norton, 2013). IS attacks target strategic data such as information exchanged by CIOs, financial data, R&D documents, customers and human resources information, etc. The consequences for organizations are manifold: deterioration of the image and brand, perturbation of the activity, financial losses, and even threatening the socio-economic, political and military ecosystems.

The financial motivation of cybercrime is obvious: whereas the annual market of data is estimated at 7B€, the revenue generated from exploiting these data is about 230B€, with an annual growth of 40% (Frenkiel, 2009). Most cyber-attacks are quite well known: denials of service, viruses, worms, Trojans, phishing, attacks on lower network layers, spywares, etc.

However, the press regularly reports unfortunate experiences. For example, in France in 2010, 150 computers of the Direction Générale du Trésor were attacked in order to capture information related to the presidency of the G8 and G20. End of 2011, the French Senate was targeted by an attack shortly before the adoption of the law on the recognition of the Armenian genocide. In July 2012, the DropBox Company acknowledged that passwords stolen on other website allowed hackers to access online stored documents. In September 2014, five million Gmail addresses and passwords were posted on a forum (Elyan, 2014).

In its annual report of 2013 (Symantec, 2013), Symantec reports that the violation of sensitive data increased by 62% with 253 incidents observed and 552 millions of identity data stolen.

During the year 2013, more than 40 million people in the U.S. had their personal information stolen, along with 54 million in Turkey, 20 million in Korea, 16 million in Germany, and more than 20 million in China.

It would be simplistic to believe that IS weaknesses are always technological, or only have an internal origin. In fact, security is a multi-faceted problem; it is as much about understanding the domain in which systems operate as it is about the systems themselves. While developing security facilities such as encryption, identity control, or specific architectures is important, our attention should be drawn at looking into the sociotechnical context in which target systems will operate and threats that may arise and their potential harm, so as to uncover security requirements.

In practice, security is still often considered as an adjustment variable (Gruselle, 2013). As the 2014 report of the (Centre d'analyse stratégique, 2014) shows it, risk management methods are still considered «too complex to be deployed in organizations, which are often not mature in terms of IS security». CLUSIF indicates that, in France, only 63% of enterprises with more than 200 employees and less than a third of Collectivités Territoriales have formalized a security policy; even though these last use EBIOS (EBIOS, 2004) and ANSSI's (Agence Nationale de Sécurité des Systèmes d'Information) PSSI (Politiques de Sécurité des Systèmes d'Information) four times more in 2012 than in 2008, this is still only limited to 13% of organizations: 63% of them do not even use a referenced method at all. The poor usage of methods is widespread: 43% of companies with more than 200 employees do not use a method or do not know which method is used to formalize the security policy. It is therefore necessary to propose simple but theoretically and empirically grounded approaches that really help organizations improve their practices, and reduce their exposure level to attacks.

2. PROBLEM STATEMENT

2.1. KNOWLEDGE IS NEEDED

Recent studies (Fenz and Ekelhart, 2009) have shown that the lack of information security knowledge at the management level is one reason for inadequate or non-existing information security management strategies, and that raising management information security awareness and knowledge levels leads to more effective strategies. In 2006, the European Network and

Information Security Agency (ENISIA) rated the establishment of unified information bases for information security risk management and the need for risk measurement methods as high priority issues.

Haley shows that among the main challenges for security projects, there is the difficulty of expressing security requirements and producing exhaustive specifications (Haley et al. 2008). In fact, most software developers are not primarily interested in, or *knowledgeable* about, security (Tondel, Jaatun, and Meland, 2008) (Mouratidis, Giorgini, and Manson, 2005), For decades, the focus has been on implementing functionalities before deadline, and patching the inevitable bugs when it's time for the next release or hot fix (Meier, 2006). However, the software engineering community is slowly beginning to realize that information security is also important for software which primary function is not related to security. Donald Firesmith claims that most requirements engineers are poorly trained to elicit, analyze, and specify security requirements (Firesmith, 2004). Consequently, they often confuse security requirements with architectural security mechanisms that are traditionally used to fulfill requirements, and end up making architecture and design decisions.

Zuccato et al. (Zuccato, Daniels, and Jampathom, 2011) report that security requirement engineering is in practice frequently performed by security non-experts. Security expertise is "scarce". Security requirements and their dependencies are often not directly known by requirements engineers.

2.2. GENERIC SECURITY KNOWLEDGE IS NOT ENOUGH, IT MUST BE DOMAIN SPECIFIC

A public library will clearly have a different view of computer security than will a central clearing house for interbank transactions. The specific security requirements of a particular installation can only be determined after careful consideration of the business context, user preferences, and/or defense posture (Devanbu and Stubblebine, 2000).

Domain knowledge, i.e the knowledge specific to the domain for which security requirements are defined, is often not taken into consideration, leaving the task to "guess" or "capture" from stakeholders domain properties to the requirements engineer or security analyst. Requirements engineers are often expected to have explicit knowledge about security and about the domain for which they elicit security requirements. However, this is unlikely to be the case; the domains are always changing, and the knowledge is too huge to understand and to master.

Some previous researches have considered knowledge-based requirements elicitation, especially with ontologies (Daramola et al., 2012a), (Velasco et al., 2009). Empirical experience with RITA (Salinesi, Ivankina, and Angole, 2008) was that "being generic, the threats in the RITA ontology are not specific to the target [bank] industry" (the case study was in the banking sector). Experts involved in the evaluation complained about the "lack of specificity of the types of threats and security requirements to the industry sector and the problem domain at hand".

2.3. KNOWLEDGE-BASED VERSUS MODEL-BASED SECURITY REQUIREMENTS ENGINEERING

A lot of research has been conducted on model-based security requirements engineering methods (Mouratidis and Giorgini, 2007) (Van Lamsweerde, 2004) (Dahl, Hogganvik, and Stølen, 2007). Model-based approaches can be useful since they provide a formal syntax for specifying security concepts. However, we believe that a knowledge-based approach can be a better option since it allows representing, accessing and re-using knowledge in order to develop methods, techniques, and tools for security requirements analysis.

Despite a considerable effort in the field (to the best of our knowledge), these issues are still not completely handled by existing approaches – in other words, most model based approaches (Mouratidis and Giorgini, 2007) (Van Lamsweerde, 2004) (Sindre and Opdahl, 2001a) and risk based approaches (Mayer, 2012) (Morali et al., 2009). The same observation can be made regarding industrial approaches' (EBIOS, 2004) (CLUSIF, 2004) lack of use of knowledge. Some knowledge based methods exist, notably the ontology-based ones (Salinesi, Ivankina, and Angole, 2008) (Velasco et al., 2009). But they do not consider the domain.

The open problem addressed in this thesis is therefore to exploit both security knowledge and domain knowledge to better guide the elicitation of domain-specific security requirements.

3. SCOPE OF THE DOCTORAL RESEARCH PROJECT

The main contribution aimed at in this PhD work is a method to "Define **domain** specific **security requirements** of an **information system** by **reusing knowledge** encapsulated in domain and security **ontologies**."

The concepts manipulated during the project belong to different research areas, notably: (a) Requirements Engineering, (b) Knowledge Engineering, (c) Security, (d) Domain

Engineering and Ontologies, (e) Information Systems. These areas of research may provide different definitions for similar concepts.

This section defines each concept as used during the PhD research project.

3.1. INFORMATION SYSTEM

There is no single definition for information systems (Carvalho, 2000). In the current PhD thesis an information system for which security requirements are elicited and analyzed is defined as follows: "*A system, whether automated or manual, that comprises people, machines, and/or methods organized to collect, process, transmit, and disseminate data (that represent user information) in a given environment"* (Longeon and Archimbaud, 1999). Regarding this definition, we join Mayer et al. (Mayer, 2012) in the fact that security of information systems encompasses not only the security of software systems or IT architecture, but also of people and facilities playing a role in the IS, so in its security. The typical example that characterizes this statement is the theft of a password of a data base through social engineering. Another example is the deliberate destruction of the data base server where the organization's information is stored by an attacker who triggers fire in the server room.

3.2. SECURITY

Literature indicates that security can have two different meanings (Firesmith, 2007)(Mayer, 2012): security and safety[1]. According to the Norwegian University of Technology and Science (NTNU): security is the protection against intended incidents. Wanted incidents happen due to a result of deliberate and planned act, while safety is the protection against random incidents. Random incidents are unwanted incidents that happen as a result of one or more coincidences (Albrechtsen, 2002). The New Oxford dictionary of English (Pearsall and Hanks, 2001) describes safety as the condition of being protected from or unlikely to cause danger, risk and injury. Security as the state of being free from danger or threat. The differences between security and safety are not remarkable; both are conditions where one is well protected. As far as this PhD is concerned, the term deals with accidental as well as deliberate harm. The basic idea is protecting assets from hazards/threats creating safe/secure conditions. This is consistent with other methods in the literature (reported in the state of the

[1] The French word "Sécurité" translates both security and safety; it is confusing for a French speaking person to distinguish between them.

art of the thesis) which don't make the distinction between security and safety and use a general definition of security as done here. Security as a discipline covers a set of methods, techniques and tools, responsible for protecting the resources of a system. As defined in ISO/IEC 13335 standard (ISO-b, 2004), in the scope of information systems, security covers seven criteria: confidentiality, integrity, availability, authenticity, accountability, non-repudiation, and reliability. The concept of vulnerability is considered as a property of the system or its environment that, in conjunction with an attack, can lead to a security failure (Anderson and Anderson, 2001). Assets are another concept in security literature defined as anything valuable in an organization (ISO-b, 2004) that might be subject to attacks (Schneier, 2003).

In this PhD thesis, we are particularly interested in security at early stages of system development, during the requirements engineering phase.

3.3. REQUIREMENTS ENGINEERING

A requirement is defined as "*a condition or a capability needed by a user to solve a problem or achieve an objective*" (Pohl, 2010). Requirements Engineering (RE) is regarded as a single phase of the software development process which main task is to develop the requirements specification for the system. RE is thus performed for each project at the beginning of the development process. The elicited requirements are documented in a project-specific requirements specification that then serves as a reference document for the following development phases (Pohl, 2010). Firesmith (Firesmith, 2007) defines requirements engineering as: "The engineering discipline within systems/software engineering concerned with identifying, analyzing, reusing, specifying, managing, verifying, and validating goals and requirements (including security-related requirements)".

In this PhD thesis, all these definitions are accepted: the requirements we intend to define are conditions stated by stakeholders to achieve their goals. Moreover, we intend to develop specifications (textual and modeling ones) that will serve as reference for later development phases. However, in our research work, we concentrate on identifying, analyzing, and reusing requirements. We do not tackle verification or validation as depicted by Firesmith. The PhD work concentrates exclusively on requirements elicitation.

3.4. SECURITY REQUIREMENTS ENGINEERING TO BETTER HANDLE SECURITY

During the last decade the research community started calling for early consideration of security, throughout the requirements engineering phase. Considering security during early stages of IS development allows IS developers to envisage threats, their consequences and countermeasures before a system is in place rather than when a destruction of possibly disastrous attack occurs (Mayer, 2012). This attention gave birth to a growing research field called Security Requirements Engineering (SRE). SRE's main concern is protecting assets from harm (Giorgini et al., 2005a).

According to (Haley et al., 2008) security requirements in the literature are defined under three categories: as functions, as non-functional requirements, as other interpretations:

A. Security requirements as functions

Security requirements are sometimes expressed by describing the security mechanisms to be used. For example, the ISO version of the Common Criteria (ISO-a, 2009), provides examples of security requirements of the general form "The [...] Security Function (TSF) shall explicitly deny access of subjects to objects based on the [rules ...]", where "rules" appear to be a mechanism. The NIST Computer Security Handbook (Guttman and Aoback, 1995) states that "These [security] requirements can be expressed as technical features (e.g., access controls), assurances (e.g., background checks for system developers), or operational practices (e.g., awareness and training)", in effect defining security requirements in terms of functions and practices.

Defining requirements in terms of function leaves out key information: what objects need protecting and, more importantly, why the objects need protecting. Both the ISO and NIST documents provide little guidance on how to connect the functionality to the security needs. Instead of describing when and why objects are to be protected, they describe how the objects are to be protected.

B. Security requirements as Non Functional Requirements (NFRs)

Devanbu and Stubblebine (Devanbu and Stubblebine, 2000) remark that security requirements are a kind of non-functional requirement. Kotonya and Sommerville (Kotonya and Sommerville, 1998), when discussing non-functional requirements, in which they include security, define them as "restrictions or constraints" on system services. Rushby (Rushby, 2001) states that: "Security requirements mostly concern what must not happen". Using the

Tropos methodology, Mouratidis et al (Mouratidis, Giorgini, and Manson, 2003a) mention that: "security constraints define the system's security requirements".

The problem with these definitions is their lack of specificity: In RE, it is not enough to say what one doesn't want. Goals and their operationalization in the context must be explicit too.

In other words security requirements should also state how "system services" are being constrained; what effect the constraint will have on the functionality of the system; and how one can validate the system against any potential constraint to ensure that it accurately reflects the stakeholders' wishes.

C. Other interpretations of Security Requirements

Several authors seem to assume that security requirements are identical to high-level security goals. Tettero et al. (Tettero et al., 1997) define security requirements as the confidentiality, integrity, and availability of the entity for which protection is needed. While this is a clear definition, in some cases it may not result in precise enough requirements.

Just like in RE in general, goals are useful to elicit requirements, analyze them, justify them, and trace them. However, requirements are goals only when it is stated that they shall be operationalized on the system and how (Van Lamsweerde, 2009)

Another term used for security requirements is "security policy". For example, Devanbu and Stubblebine (Devanbu and Stubblebine, 2000) define a security requirement as "a manifestation of a high-level organizational policy into the detailed requirements of a specific system. (Anderson and Anderson, 2001) state that a security policy is "a document that expresses [...] what [...] protection mechanisms are to achieve" and that "the process of developing a security policy [...] is the process of requirements engineering".

In this PhD thesis we make the distinction between *security criteria*, *security goals*, and *security requirements*.

Security criteria define security properties such as confidentiality, integrity, availability, and traceability. *Security goal* defines what stakeholders hope to achieve in the future in terms of security. They are at a high-level of abstraction, not always well formalized. For example a stakeholder might express the goal of maintaining the confidentiality of personal information.

Security requirements are conditions defined on the environment that needs to be fulfilled in order to achieve a security goal (they materialize a security goal) and mitigate a risk.

Consistently with Glinz's definition of NFR's (Glinz, 2007) security requirements can be non-functional requirements at a certain level of abstraction and functional at another level of abstraction. This depends on what we want to protect and on the target security level. Security requirements can be related to databases, applications, systems, organizations, and external environments. This PhD thesis also loosely uses the term 'security requirement', to refer both to 'security policy' and 'security requirement'.

3.5. WHAT IS KNOWLEDGE?

Knowledge can be defined as a justified belief that increases an entity's capacity for effective action (Kusuma T.M., 2013). According to Debenham, "Knowledge is the explicit functional associations between items of information and/or data" (Debenham, 1989).

Knowledge should be differentiated from data and information. Data (the plural of *datum*) are raw facts (Long and Long, 1998). For example; "raw signals"---- is the data. Information is the meaning we give to accumulated facts (data) (Long and Long, 1998). "SOS alert" is the meaning attached to the data 'raw signals' and it is the information.

The knowledge attaches purpose and competence to information, and it is potential to generate an action. Here: emergency SOS alert → start rescue operation (an emergency alert generates the action of starting a rescue operation)

Knowledge may be viewed from several perspectives (Kusuma T.M., 2013):

- A state of mind – knowledge is the state of knowing and understanding.
- An object – knowledge is an object to be stored and manipulated.
- A process – knowledge is a process of applying expertise.
- A condition – knowledge is organized access to and retrieval of content.
- A capability – knowledge is the potential to influence action.

Tacit knowledge is deeply rooted in actions, experience, and involvement in a specific context. It consists of cognitive element (mental models) and technical element (know-how and skills applicable to specific work). Explicit knowledge refers to knowledge that is transmittable in formal, systematic language.

Knowledge engineering is the discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise (Feigenbaum and Pamela, 1983).

Knowledge egnineering normally involves five distinct steps in transferring human knowledge into some form of knowledge based systems: (i) Knowledge validation, (ii) Knowledge representation (iii) Inferencing (vi) Explanation and justification.

A knowledge engineer is responsible for obtaining knowledge from human experts and then entering this knowledge into some form of knowledge-based systems.

It is in 1982 that Newell published a paper on "The Knowledge Level" (Newell, 1982) in which he argued the need for a description of knowledge at a level higher than the level of symbols in knowledge representation systems. The knowledge-level was his proposal for realizing a description of an artificial intelligent system in terms of its rational behavior: why does the system (the "agent") perform this "action", independent of its symbolic representation in rules, frames or logic (the "symbol" level).

In the nineties the attention of the knowledge-engineering shifted gradually to domain knowledge, in particular reusable representations in the form of ontologies. A key paper, which also quite wide attention outside the knowledge-engineering community was Gruber's paper on portable ontologies (Gruber, 1995). An ontology, being a "representation of a conceptualization" (Gruber, 1995), is a formal representation of the entities and relationships that exist in some domain. It represents a shared conceptualization in order to meet any useful purpose (Dobson and Sawyer, 2006). Ontological analysis clarifies the structure of knowledge. Given a domain, its ontology forms the heart of any system of knowledge representation for that domain. Without ontologies, or the conceptualizations that underlie knowledge, there cannot be a vocabulary for representing and reusing knowledge (Chandrasekaran et al, 1999).

In this PhD thesis, we are interested in two kinds of knowledge, the security related knowledge that is necessary to perform a security requirements elicitation and analysis, and the domain knowledge, i.e. the domain for which security requirements are elicited (e.g. bank, health, maritime, etc.). In the context of the research work, we are particularly interested in ontologies as the maine form of knowledge representation and reuse.

3.6. DOMAIN AND SECURITY KNOWLEDGE IN REQUIREMENTS ENGINEERING

A. What is meant by 'Domain'?

(Bjørner, 2010b) defines a domain as an "area" of natural or human activity, or both, where the 'area' is "well-delineated" such as, for example, for physics: mechanics or electricity or chemistry or hydrodynamics; for an infrastructure component: banking, railways, hospital health-care; or for a market: consumers, retailers, wholesalers, producers and the logistic chain.

This PhD thesis joins Bjørner in considering the domain as a universe of discourse, small or large, i.e. a structure (i) of entities, things, individuals, particulars some of which are designated as state components; (ii) of functions, over entities, which when applied become possibly state-changing actions of the domain; (iii) of events, possibly involving entities, occurring in time and expressible as predicates over single or pairs of (before/after) states; and (iv) of behaviors, sets of possibly interrelated sequences of actions and events.

B. Why is domain knowledge important for requirements engineering?

Is it possible to develop system requirements without understanding the domain? While some authors claim that this is not possible (Kaiya and Saeki, 2006) (Bjørner, 2010b), others claim that ignorance is important too (Berry, 1995). This PhD starts with the observation that analysts, consultants, designers, architects, and developers develop systems for hospitals, railways or banks with little understanding of health-care, transportation or the financial markets. This is not true in all engineering disciplines where professionalism is often ingrained: Aeronautics engineers understand the domain of aerodynamics; naval architects (i.e., ship designers) understand the domain of hydrodynamics; telecommunications engineers understand the domain of electromagnetic field theory; and so forth.

According to (Kaiya and Saeki, 2006) domain knowledge held by experts in a problem domain plays an important role on eliciting requirements of high quality. Although requirements analysts are knowledgeable on software technology, their knowledge about the problem domain where software to be developed will be operated is often tacit, not clearly defined nor formalized. Deficiency in domain knowledge may result in poor requirements elicitation.

For all these reasons, we support Bjørner in his statements: *"before software can be designed we must understand the requirements. Before requirements can be finalized we must have understood the domain".* As shown in Figure 1.1, this impacts the systems development lifecycle where domain engineering precedes requirements engineering. The results of domain engineering include a domain model: a description, both informal, as a precise narrative, and formal, as a specification. Requirements engineering include results, among others a requirements model; both informal, as a precise narrative, and formal, as a specification. The results of software design include executable code and all documentation that goes with it.

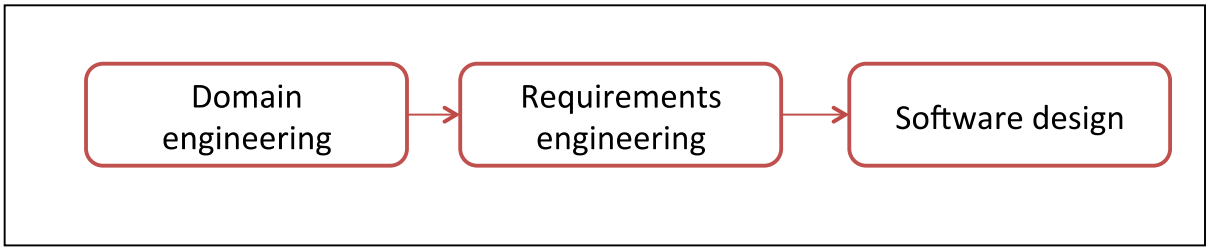


Figure 1.1. The Triptych of Software Development according to Bjorner.

C. Why is domain knowledge important for security requirements engineering?

Security requirements and security requirements engineering do not escape the observation made above. Security requirements elicitation from stakeholders is actually one of the most difficult steps in requirements analysis processes, and several methods and computerized tools have been studied and developed in order to support human activities of security requirement elicitation. However, these methods and tools are too general where problem-specific domain knowledge is not used, and did not support the utilization of domain knowledge.

Since security is not a separate issue, when it comes to define security requirements, the requirements engineer needs to know what are the organizations in the scope of the study, the assets that need protection, the different persons involved in the organization. All these information falls within the domain at hand. Moroer, the countermeasure to overcome the risks, differ weather the organization belongs to the health domain, banking, aeronotics, etc.

knowing what are thr countermeasures specific to each domain will help elicitating security requrements of a better quality.

3.7. KNOWLEDGE REUSE IN SECURITY REQUIREMENTS ENGINEERING

Back in 1993, the second International Workshop on Software Reusability was held in Lucca, Italy. Most of the papers presented at this event focused on reusing code, design or architecture. In other words, the thinking was that mainly the hard artifacts—code, object, and so on—could be reused. Very few papers looked at the idea of reuse earlier in the IS life cycle, namely reusing requirements themselves. Active areas of reuse research in the past twenty years include reuse libraries, domain engineering methods and tools; reuse design, design patterns domain specific software architecture, software componentry, generators, measurement and experimentation (Frakes and Kang, 2005).

Nowadays, the practice of reuse is moving upstream and reuse is also concerned with more abstract artifacts. Requirements are commonly recycled; patterns are exchanged on the Internet. The notion of reuse at the requirements stage is largely accepted by many within the community as a desirable aim (Lam, McDermid, and Vickers, 1997). For instance, a working conference on patterns (Pattern Languages of Programs) is held twice a year and results in the sharing of knowledge and publication of new patterns (Robertson and Robertson, 2013).

Requirement reuse can be defined as either taking requirements that have been written for previous projects and then using them for a new project, or writing requirements from scratch at a reasonable level of generality and abstraction in order to use them over different projects. A clear distinction must be made between engineering “for” reuse and engineering “by” reuse (Maiden, 1993).

For instance, it is possible to reuse different types of data, ranging from business requirements and functional requirements to use cases and test cases. Since requirements engineering is the first phase in the software development process, requirements reuse can empower the software life cycle. Previous research (López, Laguna, and Peñalvo, 2002) has pointed out that reusing the first software products and processes implemented in a development project can have an impact on the life cycle from two basic points of view: (a) allowing the software development resources to be more profitable, and (b) promoting reuse-based development across the entire software process.

Moreover, at a high level of abstraction, every application tends to have the same basic kinds of vulnerable assets (e.g., data, communications, services, hardware components, and personnel). Similarly, these vulnerable assets tend to be subject to the same basic kinds of security threats (e.g., theft, vandalism, unauthorized disclosure, destruction, fraud, extortion, espionage, etc.) from attacks by the same basic kinds of attackers (e.g., hackers, crackers, disgruntled employees, international cyber terrorists, industrial spies, etc.) who can be profiled with motivations and their typical levels of expertise and tools. Security requirements tend to be even more standardized than their associated mechanisms. For example, to address the identification and authentication requirements, one may have several choices of architectural mechanisms beyond user ID and passwords (Firesmith, 2004). Based on this, requirements reuse could lead to significant savings in development time and cost (Hermoye, van Lamsweerde, and Perry, 2006).

Structuring security knowledge helps the knowledge consumer to browse the content and to find the relevant information more efficiently. Different knowledge representations exist in the literature. Patterns of recurring attacks and vulnerabilities have been identified by longtime software security practitioners (Mead and Stehney, 2005). Security templates of a high level of abstraction were also introduced for reuse purposes (Firesmith, 2004). Various other approaches for managing security knowledge and reuse exist in the literature, such as taxonomies, ontologies, standards, and guidelines.

During the last decade, researchers paid some attention to the benefits of reuse in SRE process given the common nature of security problems across applications and application domains (Firesmith, 2004). Security knowledge is hard to acquire. In addition to awareness about potential attacks, designing security-critical systems requires knowledge and security expertise in various fields such as computer networks, operating systems, communication protocols, cryptography algorithms, and access control methods. Reuse combined with predefined structured knowledge can make the job of requirements engineers much easier and faster, since they usually lack security expertise and skills. However, one should be careful when structuring reusable knowledge – it has to be of a high quality. Otherwise it might end up introducing new security problems.

In this PhD thesis, security knowledge includes security threats and attacks, vulnerabilities, security requirements. Domain knowledge includes domain entities such as actors, organizations, assets, locations, etc.

4. RESEARCH QUESTIONS

The thesis addresses the aforementioned problems by proposing a method that guides requirements engineers (but also security analysts) in the elicitation of security requirements for specific domains. The method is generic in the sense that it can be applied to various domains but it is domain specific at its application.

Thus, the main objective of this thesis is to reply to the following research question:

Main research question: *How to elicit security requirements that are domain specific with a reuse strategy?*

To answer this main research question, several sub-issues must be considered too. Addressing each of the following four research questions is necessary to solve the main research question of the thesis:

RQ1. How to structure security knowledge both on a generic level and in relationship with specific domains?

RQ2. How to structure the knowledge related to domains in the light of security concerns?

RQ3. What is the best way to capture stakeholders' security goals and requirements?

RQ4. How to exploit both security knowledge and domain knowledge to produce domain specific security requirements and build security requirements models?

5. RESEARCH HYPOTHESES

The main purpose of the research work in this PhD is the elicitation of domain specific security requirements through a reuse strategy. Different approaches can be adopted to accomplish this objective. One possible approach could be to derive from generic security knowledge, domain specific security knowledge, and then use this domain specific security knowledge for domain specific security requirements elicitation. Another approach could be to use both security knowledge and domain knowledge to elicit domain specific security requirements. In this PhD thesis, the second approach is developed. However, within this approach, questions arise concerning the best structure of the security and domain knowledge, and how to use this knowledge to build security requirements models.

To summarize, the PhD work tests the main hypotheses:

a. It is possible to provide the necessary security knowledge by choosing adequate structures for it;
b. It is possible to build security requirements models by relying on security and domain knowledge;
c. Relying on domain knowledge in addition to security knowledge will make a difference in the production of domain specific security requirements with regards to relying only on generic security knowledge;
d. The latter approach will be efficient and easy to use for end users;
e. A good implementation of the last approach is possible and useful for end users.

6. RESEARCH METHOD

The research work reported in this thesis was conducted using a design science method. (Hevner and Chatterjee, 2010) define design science as being about design and validation of solution proposals to practical problems. Hevner et al. (Hevner et Chatterjee, 2010) suggest that design science differs in two aspects from other branches of science: (a) it is concerned with artifacts rather than facts of nature or social structure, and (b) it is concerned with a search for prescriptive rules for design, rather than a search for descriptions, explanations and predictions, as other branches of science are. Being design oriented, the research method used in this PhD thesis intends to validate the research hypotheses presented above by means of case studies and controlled experiments.

To reply to the main research question and to test the research hypotheses of this thesis, we carried out a research strategy based on the design science process model proposed by (Peffers et al., 2007). This process contains six main steps: (identify problem and motivate, define objective of a solution, design and development, demonstration, evaluation, communication). Figure 1.2 presents the design science process model for information system research, and the application of this process to the research carried out in this PhD thesis.

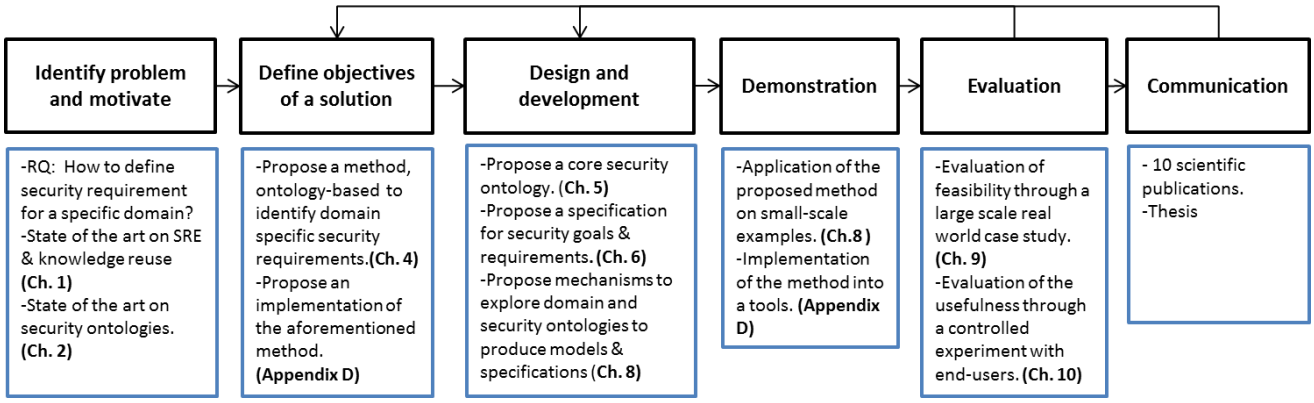


Figure 1.2. Application of the design science process model for information system research (Peffers et al. 2007) to the research carried out in this thesis.

More precisely, the following strategy was implemented:

A. A systematic mapping study on the reuse of knowledge in security requirements engineering was conducted to (i) update existing literature surveys related to SRE with recent researches, (ii) to identify the (re)used knowledge in SRE, (iii) to distinguish between the different types of knowledge reuse structures in SRE, and (iv) to understand their use, and (v) to investigate and report the definition of domain specific security requirements by SRE methods.
B. A survey of security ontologies was conducted. The main objective was to review, analyze, select, and classify security ontologies, as a scope study but with a particular interest in the field of security requirements engineering.
C. A collection of gaps and drawbacks in the existing approaches were identified with regards to the research question of this PhD thesis.
D. An ontology-based method was proposed to analyze and elicit domain specific security requirements. Within the proposed method, three sub-propositions were developed: core security ontology, specification of security goals and security requirements, and a rule-based technique to explore domain and security ontology during the requirements elicitation process.
E. The feasibility of the proposed method was evaluated through a real-world case study. The ease of use and usefulness of the proposed method were evaluated with end-users. The results of these evaluations were intended to support or refute the hypothesis proposed in this thesis.
F. New research directions were identified. The results of the case study and the controlled experiment and the feedback from the information system and computer science community were taken into account to improve the initial method.

7. CONTRIBUTIONS

This PhD thesis makes some original contributions:

- A framework for analyzing and comparing knowledge reuse in SRE. The framework was proposed after a systematic mapping study of knowledge reuse in security requirements engineering. The systematic mapping study analyzed more than 150 publications over the past twenty years. The different reuse forms in security requirements engineering (ontologies, patterns, taxonomies, generic models,..) were identified, the reuse techniques, as well as the reused knowledge. This work published in the requirements engineering journal (REJ).
- A framework to classify security ontologies. In this contribution, 8 families of security ontologies were identified after a deep reading of about 50 publications. The 8 families are beginning security ontologies, security taxonomies, general security ontologies, specific security ontologies, web oriented, risk based, modeling, and for requirements. Each of the security ontologies of the study was analyzed in terms of its overage of security aspects. This contribution has been presented in the third International Workshop on Information Systems Security Engineering WISSE'12.
- An ontology based and a rule based method for domain specific security requirements elicitations. The method takes as an input a core security ontology and any domain ontology to produce domain specific textual security requirements as well as security models. This method is based on four main sub-contributions:
    - A core security ontology. This contribution proposes a novel meta-ontology for security requirements engineering. It combines elements of existing ontologies to create a core security ontology. A description of the core security ontology itself is proposed and the process for constructing it. Several elements of the ontology are defined and their relations are shown. An evaluation is presented in order to prove the completeness, the validity and the usability of the ontology. This contribution was presented in the International Symposium on Engineering Secure Software and Systems in March 2015.
    - A multi-level domain ontology to structure domain knowledge. This multi-level structure allows the method to avoid being stacked to a single domain. The advantage is to give the possibility to switch from a domain to another while using the method.

- Linguistic templates to specify security goals and security requirements. Here, the differences between security goals and security requirements are clarified. Linguistic templates are proposed to specify them.
- A rule-based mechanism to extract relevant knowledge from domain and security ontologies and produce security requirements models and textual SRE specifications.

The method has been evaluated. Its feasibility was evaluated through a real world case study. Its usage and usability were evaluated through a controlled experiment. Different parts of the method have been published and presented in various events and editions. One of the publications got a best paper award. Here follows the list of publications:

- Vers une nouvelle génération de définition des exigences de sécurité fondée sur l'utilisation des ontologies. **Best paper award** (INformatique des Organisations et des Systèmes d'Information et de Décision - INFORSID 2012, Montpelier, France).
- Une méthode de définition des exigences de sécurité fondée sur l'utilisation des ontologies. (Séminaire Doctoral du Forum Académie - Industrie de l'AFIS., 2012, Paris, France.)
- Towards a new generation of security requirements definition methodology using ontologies (Conference on Advanced information Systems Engineering, CAiSE 2012 Doctoral Consortium, Gdansk, Poland)
- Using Security and Domain ontologies for Security Requirements Analysis. (Computer Software and Applications Conference Workshops – COMPSACW 2013, Kyoto, Japan).
- Security requirements analysis based on security and domain ontologies. (Requirements Engineering Foundation for Software Quality – REFSQ 2013, Essen, Germany).
- A Methodology for Defining Security Requirements using Security and Domain Ontologies (Insight Journal, INCOSE - The International Council on Systems Engineering, 2013).

The full list of publications produced by this PhD can be consulted in the appendix G.

8. ORGANIZATION OF THE THESIS

This thesis is organized in three parts and ten chapters as follows:

Part 1: State of the art

The first part presents the state of the art and contains 2 chapters:

> **Chapter 2** reviews related work presented in literature regarding security requirements engineering and knowledge reuse. This chapter tackles seven research questions about the state of the art on the reuse of knowledge in SRE about the knowledge reliance, its form of representation, the reusable knowledge, the technique for reusing the knowledge, the domain knowledge reliance, gaps and challenges found in the literature. These questions were answered through a systematic mapping study presented in the chapter.
>
> **Chapter 3** reviews related work presented in the literature related to security ontologies. The chapter raises three main questions about the different security ontologies that exist nowadays, about their coverage of security aspects, about the 'best' security ontology that one can choose for the definition of IS requirements.

Part 2: Main contribution

The second part of the PhD thesis presents the main body of the contribution. It contains five main chapters:

> **Chapter 4** provides an overview of the domain specific security requirements elicitation method. It presents briefly the main elements on which the method relies. The chapter introduces a motivating example that will be used in the rest of the thesis to illustrate our method.
>
> **Chapter 5** presents the first contribution of this PhD thesis: a core security ontology for security requirements elicitation. The ontology is presented. Its concepts and relations are described. The completeness of the ontology is validated with regards to other security ontologies. The validity of the ontology is validated through some indicative queries.
>
> **Chapter 6** describes security goals and security requirements. It presents the security goal model and the syntactic security requirement pattern proposed to handle security goals and requirements in the method.

Chapter 7 describes the multi-level domain ontology that was proposed to encompass the domain knowledge. The ontology allows switching from a domain to another through its upper domain ontology.

Chapter 8 presents how the previous elements (security ontology, security goals and requirements, multi-level domain ontology) were put together in the method to produce security goals and textual specifications. The chapter presents an algorithm and the necessary proposed rules (mapping rules and production) to extract the relevant knowledge from the ontologies. In the chapter the process is presented step by step through the running example.

Part 3: Evaluation

The third part of the PhD thesis reports the evaluation of the proposition. It contains two chapters:

Chapter 9 presents the evaluation of the feasibility and genericity of the method. It reports the application of the method on a real world case study related to the maritime domain. The case study demonstrates the capacity of the method to handle security requirements engineering in a specific domain of application. The chapter examines the genericity of the method through two other applications.

Chapter 10 presents the evaluation of the ease of use and usefulness of the method. This is done thanks to a controlled experiment with end-users. The chapter reports the obtained results and discusses the feedback gathered from participants with regards to the hypothesis set in the beginning.

Part 1

State of the art

Chapter 2

Security requirements engineering and knowledge reuse

An increasing number of publications, conference tracks, and workshops in recent years point out the growing interest of researchers and practitioners in providing SRE processes with various frameworks and methods. Some of them are extensions of goal-oriented approaches, like Secure i* (Liu, Yu, and Mylopoulos, 2002), Secure Tropos (Mouratidis, 2006), KAOS and anti-models (Van Lamsweerde, 2004). Others are built on the object paradigm, mainly UML-based, such as misuse cases (Sindre and Opdahl, 2005), security use cases (Firesmith, 2003), Secure UML (Lodderstedt, Basin, and Doser, 2002) and UMLSec (Jürjens, 2002)(Jürjens, 2005)(Jürjens and Shabalin, 2004).

Nowadays, the research community in SRE as well as practitioners has a vague idea of existing literature for handling knowledge reuse among existing SRE approaches. Despite some existing surveys about security requirements engineering, there is not yet any reference for researchers and practitioners that presents in a systematic way the existing proposals, techniques, and tools related to security knowledge reuse in security requirements engineering. The definition of security requirements for specific domains is still an issue (Salinesi, Ivankina, and Angole, 2008)(Souag, 2012), and no reference reports in a systematic way the situation of domain consideration in SRE.

For instance, a quick research indicates that some approaches propose a catalog of attacks (Jensen, Tøndel, and Meland, 2010), while others rely on patterns (Hatebur, Heisel, and Schmidt, 2007). However, a systematic mapping study and analysis of existing security requirements engineering methods that make (re)use of knowledge is still lacking.

This chapter presents a structured and systematic mapping study (Petersen et al., 2008) of several articles related to knowledge reuse and security requirements engineering from the last two decades.

This mapping study must find answers to the following questions: Does the security requirements engineering method rely on reusability of knowledge? What are the reusable elements? How are they represented, modeled? How are they (re)used? Are the knowledge based approaches tool-supported? Are the security requirements defined for specific domains? Which domains? How is the domain knowledge represented?

A framework was defined to understand the different proposals and classify new contributions in the future. Over 100 papers were analyzed from which the chapter reports the knowledge reuse situation of 30 methods.

The chapter is structured as follows. Section 2 gives an overview of our research method. Section 3 presents the process and results of the conducted systematic mapping study to get an overview of existing knowledge reuse based security requirements approaches. Section 4 summarizes the results and answers the research questions. Section 5 reports the related works. Section 6 discusses threats to validity of our mapping study. Finally, Section 7 concludes this chapter.

1 RESEARCH METHOD

A fair amount of publications, conference tracks and workshops in SRE appeared during the last decade, revealing a steady interest of both researchers and practitioners in that domain. Unfortunately, it remains difficult to have more than a vague idea about what is available in terms of reuse of security requirements, and to position research with respect to available practices in order to choose appropriate practice. One difficulty is due to the fact that these issues are addressed by several communities: the requirements engineering community, the software engineering community, the information systems community, and the computer security community.

Our research method aimed at analyzing and identifying the available literature on security requirements research, and categorizing it in a systematic way. The Systematic MAPping study (SMAP) was conducted between August 2013 and December 2013. We applied the mapping studies guidelines proposed by Petersen et al. (Petersen et al., 2008), which compare the methods used in mapping studies and systematic literature reviews. The SMAP reported in this chapter was performed based on these guidelines (cf. Figure 2.1), to identify questions and answers rose by the research community on knowledge (re)use in SRE.

Reviewing existing research in a fully objective way is not possible. A systematic study, such as the one outlined in Figure 2.1, however makes the process less subjective by using pre-defined data forms and criteria that narrow the scope for personal interpretation.

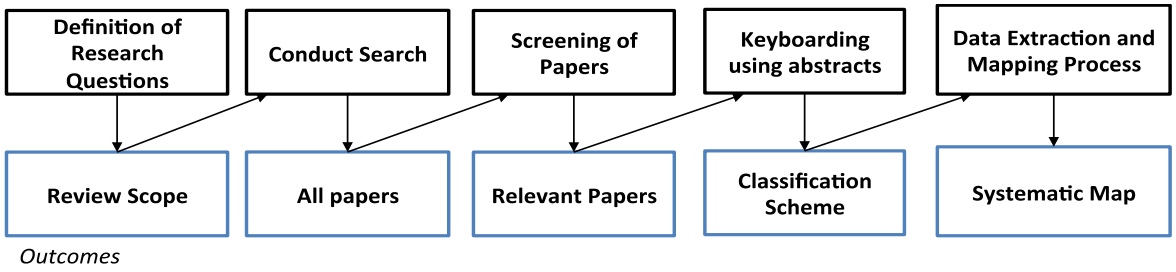


Figure 2.1. The systematic mapping process carried out in this paper, applied from (Petersen et al. 2008)

Mapping studies must be distinguished from systematic literature reviews in several ways. Systematic Literature Reviews (SLR) have been defined as "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest" (Kitchenham and Charters, 2007). Mapping studies are a special kind of SLR that use the same basic methodology as SLRs but aim to identify and classify all research related to a broad software engineering topic rather than answering questions about the relative merits of competing technologies as addressed by conventional SLRs (Kitchenham, Budgen, and Brereton, 2011) (Budgen et al., 2008). SMAPs are intended to provide an overview of a topic area and identify whether there are sub-topics with sufficient primary studies to conduct conventional SLRs and also to identify sub-topics where more primary studies are needed.

Overall, the main phases of our systematic mapping study were: the definition of research questions, conducting the search for relevant papers, screening of papers, key wording of abstracts and data extraction and mapping planning, conducting, and reporting. Figure 2.1 presents the process structure of our SMAP.

A key element in the guidelines proposed by Petersen et al. (Petersen et al., 2008), is the definition of the *research questions (research scope).* Research questions should reflect and reply to the main goals of a SMAP in providing an overview of a research area, identify the quantity and type of research and results available within it. The search for *primary studies (all papers)* is conducted by using search strings on scientific databases or browsing manually through relevant conference proceedings or journal publications. Screening papers for inclusion and exclusion *(relevant papers).* In this step, restriction and quality criteria are used to exclude studies that are not relevant to answer the research questions. *Key wording*

using abstracts is a way to reduce the time needed for developing the *classification scheme* and ensuring that the scheme takes the existing studies into account. The process ends up with the *data extraction and mapping of studies;* here the classification scheme evolves by adding new categories or mapping and splitting existing categories. More practical details on how we addressed these issues in our SMAP are given in the next section.

2 MAPPING STUDY: REUSABLE SECURITY KNOWLEDGE

The review includes publications reporting on existing approaches and tools as well as publications discussing research issues for security requirements and knowledge reuse in SRE. The SMAP was conducted in 24 relevant sources (The detailed list of the sources can be consulted in the cell "Digital library/resource" in Table A.1 in the appendix A). The total retrieved number of publications is **158** using well-defined search criteria (which will be presented later). From these 158 publications, **95** papers were chosen for further analyses based on our set of selection criteria. The complete list of all 95 retrieved publications and details about the retrieved searches can be found in the Appendix A.

2.1. CONDUCTING THE SYSTEMATIC MAPPING STUDY

The main goal for conducting the systematic mapping study was i) to get an extensive overview of existing knowledge based approaches and tools for security requirements engineering and ii) to understand key issues for security requirements elicitation and analysis considering the (re)use of knowledge in these practices. This systematic mapping study was developed using the following elements (definition of research questions, search for primary studies, screening of papers, data classification):

A. Definition of research questions

Security requirements engineering process needs to be reinforced with knowledge on security. This knowledge is necessary to take into account security requirements early and consistently. Hence, we wanted to understand the current state-of-the-art in this field. More specifically, we wanted to evaluate if the security knowledge can be reused (RQ1). A deep analysis of this question requires that we elicit how this knowledge is represented (RQ2) and reused (RQ3). Moreover, can the whole knowledge be reused (RQ4)? Can it be reused automatically (RQ5)? We were also interested if domain knowledge was used and if the

produced security requirements are domain specific (RQ6). Finally, what can be improved in current approaches (RQ7)? RQ2, RQ3, RQ4, RQ5 and RQ6 only make sense if RQ1 is answerable and if the answer is yes. RQ2, RQ3, RQ4, RQ5 and RQ6 are necessary to understand which knowledge is currently reused and RQ7 sketches avenues for future research. Thus, the systematic mapping study was guided by the following research questions, for each SRE method, and for SRE methods overall:

RQ1. Does the security requirements engineering method rely on reusable security knowledge? How many papers handled knowledge reuse in SRE? **(Knowledge reliance)**

RQ2. How is the reused knowledge represented? What are the proportions of each knowledge representation form? **(Form of representation)**

RQ3. What are the techniques for (re) using the knowledge and their proportion? **(Technique)**

RQ4. What are the main reused elements and their proportion? **(Reusable knowledge)**

RQ5. Is it tool-supported? Are there many tools for SRE overall? **(Automation)**

RQ6. Are the security requirements specified for a specific domain? If yes, for which domain? And how are they represented? **(Domain)**

RQ7. What are the new challenges regarding security knowledge (re) use in SRE?

Research Question RQ1 checks, among the different existing proposals, whether the security engineering method at hand relies on the (re)use of knowledge. It also looks for the number of papers that rely on the (re)use of knowledge. RQ2 finds how (and how much) the (re)used knowledge is represented (modeling language, representation of requirements, etc.). RQ3 identifies how the security knowledge is (re)used. RQ4 reports what the main reusable elements found in proposals identified in RQ1 are: for instance, security requirements, threat models, or common vulnerabilities. RQ5 checks whether the SRE method offers automated support for the reuse of knowledge. It also examines the number of papers that propose tools for reuse in SRE. RQ6 checks if the security requirements analysis addresses specific domains, Finally, RQ7 extracts from the papers some new challenges that the SRE community should face in the future.

B. Search for primary studies

To *search for primary studies (all papers)*, the sources (presented in the Appendix A) were selected based on an analysis of security requirements literature. Our sources were extracted from digital libraries such as ACM Digital Library, Science Direct, IEEE Xplore, IEEE Computer Society, SpringerLink and DBLP, those digital libraries were chosen because the institution (Paris 1 Panthéon Sorbonne) had a subscription to them. Also journals, conferences, and workshops of the domain such as RE, REFSQ, ARES, Requirements Engineering Journal were considered. These sources were chosen based on a pre-search on Google Scholar in addition to consulting the citations of existing SLRs and other SMAPs. Relevant books and reports were explored further. For all primary studies found in these sources we also followed their relevant cited references to find additional contributions outside the above-mentioned subset. All searches have been conducted on publications appeared between 2000 and 2013, thus covering over 13 years of SRE research.

Depending on the source, different search terms were used. For the more general conferences and for journals we used the search terms "reuse security requirements", "knowledge security requirements", "reusability in security requirements" or "knowledge reusability security requirements" appearing in the full-text of the publications (excluding references). In conferences and journals related to SRE, the search term was iteratively refined, for example leading to the search terms "ontologies for security requirements", "pattern security requirements", "reuse misuse cases", "knowledge security use cases" or "reuse secure Tropos".

C. Screening of papers

Search for primary studies lead to 158 articles, many of which were irrelevant. Screening for papers based on the title and succinct review of the abstract, in addition to the reliance on our inclusion and exclusion criteria, reduced the number of relevant papers. The screening process was performed by me and validated by my supervisors.

The following restrictions and quality criteria for including/excluding publications were defined:

- *(Restriction R1)* The study only includes papers available in electronic form. Books were analyzed based on information available online and using the hard copy versions.
- *(Restriction R2)* Only publications written in English were included.

- *(Quality criterion Q1)* Each publication was checked for completeness. Publications containing several unsupported claims or frequently referring to existing work without providing citations were excluded.
- *(Quality criterion Q2)* Articles related to the topic of this paper published between 1st January 2000 and 31st August 2013 were included: i) papers proposing any method for SRE; ii) papers proposing knowledge reuse based methods for SRE; iii) papers proposing automation of any (knowledge reuse based) SRE.
- *(Quality criterion Q3)* Works of the same authors with very similar content were included and grouped under the same category (method).
- *(Quality criterion Q4)* Some articles were intentionally excluded to keep the level of the SMAP manageable, in particular when the proposition was not relevant enough to the topic of our research.

Ninety-five searches in 24 sources were carried out using the search terms described above. In total 158 publications were retrieved, out of which 21 were found not directly in the 24 sources but by following relevant cited references. Figure 2.2 shows the distribution of research results related to security requirements engineering and reuse between 2000 and 2013. The figure also shows that, between 2004 and 2007, a great number of publications were published; thus, the well-known approaches for security requirements engineering appeared during this period. Table A.1 (in the Appendix A) presents the retrieved and selected publications for each source.

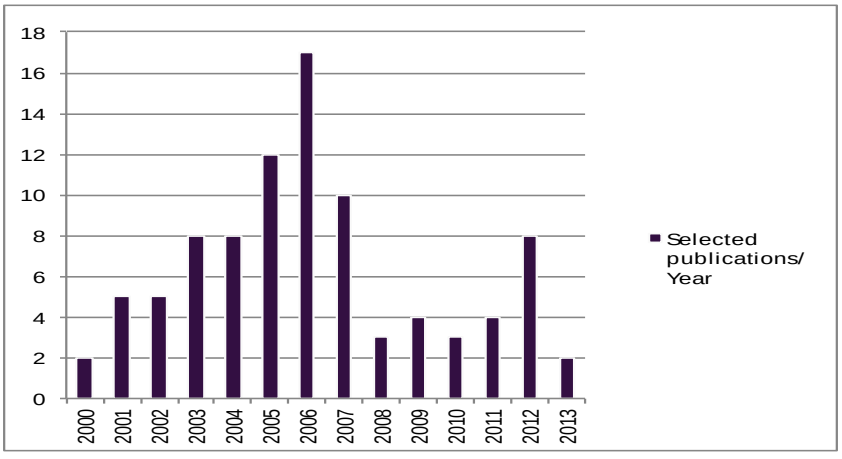


Figure 2.2. Number of selected publications on (knowledge-based) security requirements engineering (2000–2013).

D. Data classification

The retrieved publications were first analyzed regarding the restrictions R1–R2. The remaining publications were carefully assessed regarding quality criterion Q1. For each retrieved publication the following standard information was collected in a data extraction form:

- Date of search, source, and used search term.
- Authors, title, and publication year.
- Type of publication (conference, workshop, journal, report, or book).
- Short summary (main claims, presented approach/tool).
- Restrictions R1, R2 (yes or no)?
- Quality criteria Q1, Q2, Q3, and Q4 fulfilled (yes or no)?
- Addressed research question(s).
- Selected (yes or no)? Based on restrictions and quality criteria.
- Comments/rationale regarding selection.
- Need for tools. Does the publication stress the need for support (yes or no)?

For each selected publication the following additional information was captured in a second form to increase confidence regarding their relevance for security requirements engineering elicitation and analysis: the main focus of the publication is on security requirements (yes) vs. security requirements are only addressed as part of a broader approach (no)?

Searching the security requirements approaches and (re)usable knowledge based security requirements approaches conferences led to 158 papers, out of which 95 (60%) were related specifically to security requirements approaches. Among these 95 papers, 29 papers (31%) addressed reuse of knowledge for security requirements. Searching conferences led to the largest set of results: 39 papers (41%) out of 70 papers found. Note that the highest number of papers presented in conferences was found in two main conferences: The international conference on Availability, Reliability and Security (ARES) with 13 papers found out of which 8 were selected; and the international conference on Requirements Engineering (RE) with 24 conference papers found and 7 selected.

The number of selected journal papers was 20 (21%) out of 31 papers found. The total number of workshop papers found was 21, out of which 11 (12%) were selected. 15 (16%)

relevant technical reports were also considered in our search. Books and book chapters were taken in consideration too: out of 18 retrieved sources, 10 (10%) were selected.

Table A.1 in the appendix A gives all the details about the retrieved publications, their types and the ones selected. Figure 2.3 summarizes the statistical results of all selected papers by categories (books, conferences, workshops, reports).

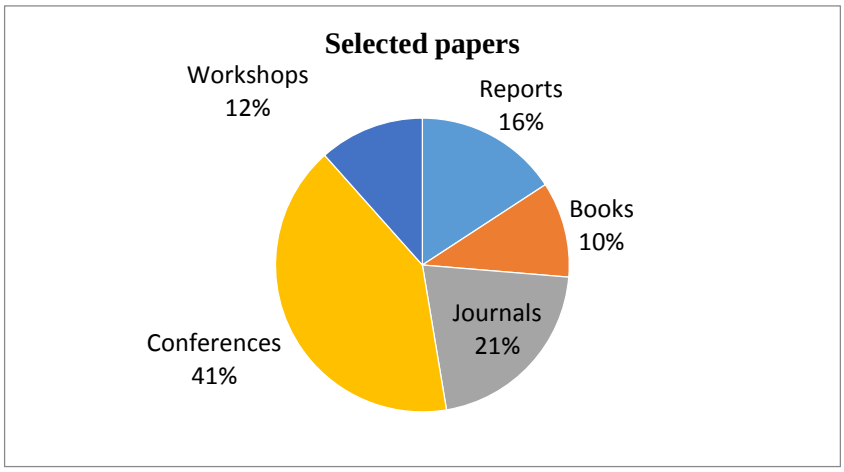


Figure 2.3. Relative share of the various paper types in the selected set

For the selected papers, we were also interested in the type of the research. As recommended by (Petersen et al., 2008), we adapted the classification system developed by (Wieringa et al., 2006) for requirements engineering paper classification. The papers were thereby classified into:

- Solution proposal: papers that discuss new or revised techniques,
- Philosophical: papers that sketch a new way of looking at things, a new conceptual framework, etc.
- Evaluation research: papers that investigate a practice or an implementation in practice and report the lessons learned.
- Validation research: papers that investigate the properties of a solution proposal that has not yet been deployed in practice.
- Opinion papers: papers that contain the author's opinion about a research or practice subject.
- Experience: papers that are often from industry practitioners or researchers who have used their tools in practice. They report how something was done in practice.

Along our classification, some papers covered two categories. For example, a paper may be at the same time a "Solution proposal" and "Validation research". In such cases, we labeled

them (Solution proposal + Validation). Conversely, some papers could not be linked to any category since they were exclusively describing tools. Such papers are of interest for us practically and address one of our research questions. Thus, we decided to use the label "Tool".

Table 2.1 summarizes the results of the classification. Most of the papers are solution proposals (41%), few of which are validated (22.1%). Evaluation researches that investigated the practices in industry are only (10.5%). Eight papers were exclusively presenting tools.

Table 2.1. Type and number of selected papers

	Solution Proposal	Philosophical viewpoint	Evaluation	Validation	Experience	Opinion	Tool	Solution + Validation + Tool	Solution+ Validation
Book (chapter)	5	2	1	0	0	0	2	1	0
Journal	10	0	2	0	1	0	0	2	5
Conference	15	2	4	0	1	0	5	0	10
Workshop	4	2	1	2	0	0	0	2	1
Report	5	2	2	0	0	0	1	0	5
Total	**39 (41%)**	**8 (8.4%)**	**10 (10.5%)**	**2 (2.1%)**	**2 (2.1%)**	**0 (0%)**	**8 (8.4%)**	**5 (5.26)**	**21 (22.1%)**

3 A FRAMEWORK FOR ANALYZING AND COMPARING KNOWLEDGE REUSE IN SRE (DATA EXTRACTION AND MAPPING OF STUDIES)

Extracting the data, while surveying in depth the different approaches for SRE with regards to knowledge reuse, allows us to define the different categories covered by the study and construct the map (i.e. a framework for analyzing and comparing knowledge reuse in SRE). The framework shown in Figure 2.4 is structured around facets that capture individual dimensions related to knowledge reuse in SRE. This framework makes it possible to organize the different methods, techniques and tools for knowledge reuse in SRE around different axes that were identified through the SMAP and appeared relevant to us.

3.1. KNOWLEDGE

The knowledge facet identifies the different knowledge (re)used in SRE. This facet was organized under three main sub-dimensions (by analogy to the classification framework proposed by Dubois et al. (Dubois et al., 2010)):

- Organization & Assets: all the knowledge related to the organization, its assets, its actors can be (re)used over different projects.
- Risk: knowledge related to risk addresses different threats that might threaten the assets of an organization, the vulnerabilities that might be explored, the attackers and their attack methods.
- Risk treatment: knowledge related to mitigating the risk, such as security requirements, countermeasures, security policies, etc.

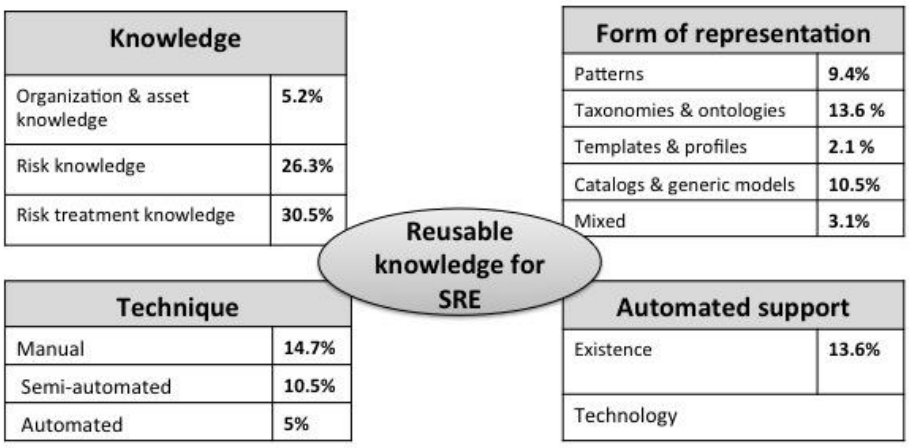


Figure 2.4.Framework for knowledge reuse in SRE.

3. 2. FORM OF REPRESENTATION

The "form of representation" facet indicates the different types of knowledge forms that were identified and how they are organized: patterns, taxonomies and ontologies, templates and profiles, catalogs and generic models, mixed.

3.3. TECHNIQUE

The technique facet is about the knowledge (re)use techniques. These can be automated (e.g. queries), semi-automated (e.g. process), or totally manual (e.g. guidelines).

3.4. AUTOMATED SUPPORT

The "automated support" facet checks the existence of an automated support for knowledge (re) use in SRE, and its technology features.

The next section details the publications retrieved, and replies to the research questions relying on the presented framework.

4 DETAILS OF THE SYSTEMATIC MAPPING STUDY

Table 2.2 presents an overview of the security knowledge reuse in SRE methods. Columns contain the main concepts characterizing the conceptual space of security. Lines cover the different reuse forms by SRE methods. Cells contain a colored area when there exist SRE publications proposing a reuse based approach of a given reuse form, for a given security concept. The colors of the cells get darker according to the number of publications covering it. It is white when there is no publication describing such a link.

This presentation should help the reader to understand the security reusable knowledge in the body of literature. It also helps to retrieve for each concept of security (e.g. security requirement) how (through ontologies, templates) and how much it is reused. As an illustration, the security concept 'threat' is covered by a lot of publications proposing to reuse it through ontologies or taxonomies.

Table 2.2. Security knowledge reuse in SRE

	Organization	Asset	Threat	Vulnerability	Security goal	Security requirement	Counter-measure
Ontologies/ taxonomies							
Catalog/ generic models							
Patterns							
Templates							
Mixed							

The following paragraphs go into the details. They present a brief description of the SRE method, followed by answers to the research questions (method, form of representation,

technique, and automation). The paragraphs report the different types of reusable elements and how they are modeled and described, and how they are used.

In fact, there are different ways for presenting and classifying SRE methods, depending on the angle from which we study and analyze them. For instance, Fabian et al. (Fabian et al., 2010) organize SRE methods into 6 main categories (multilateral, UML-based, goal-oriented approaches, problem frames-based, risk analysis-based, common criteria-based). Elahi (Elahi et al., 2011) organizes SRE methods into two main categories depending on whether the method focuses on the threats and vulnerabilities (the dark side of security) or on security requirements and countermeasures (the white side of security).

As the main goal of this chapter is to focus on knowledge (re) use in SRE, the different methods will be presented using the result of RQ2, i.e. according to the knowledge (re)use form used. Thus, we distinguish: methods that reuse patterns, taxonomies and ontologies, catalogs or generic models, mixed forms of reuse. We also distinguish methods that do not reuse any kind of security knowledge.

4.1. METHODS THAT (RE) USE SECURITY PATTERNS

A security pattern describes a particular recurring security problem that arises in specific contexts and presents a well-proven generic scheme for its solution (Mouratidis et al., 2003). The SMAP found that some SRE methods (re) use patterns during the SRE process in the form of models or in other forms.

A. Patterns of models

The identified SRE that (re) use patterns of models are presented below:

- *KAOS and Anti-Models:*
  (Description) Lamsweerde (Van Lamsweerde, 2007)(Van Lamsweerde, 2004) extended the KAOS method to support security issues at the requirements level. KAOS is a requirements engineering method dealing with the elaboration of the objectives to be achieved by the system-to-be, the operationalization of such objectives into requirements and assumptions, the assignment of responsibilities for those specifications to agents such as humans, devices or software, and the evolution of such requirements over time and across system families.

  (Knowledge reliance) Hermoye et al. (Hermoye, van Lamsweerde, and Perry, 2006)

enriched the KAOS framework with an attack pattern library and reusable countermeasures built after analyzing commonalities in goal-oriented specifications from some case studies. (**Reusable Knowledge)** In this approach, a reusable attack pattern captures common objectives of malicious agents for known attacks (e.g., replay, denial of service, password attacks). Reusable countermeasures are reusable anti-goal resolutions. For example, countermeasures against replay attacks may include freshness mechanisms. **(Form of representation)** Hermoye et al. use a library of attack patterns; an attack pattern is a fragment of an anti-model defined on an abstract domain. Attack patterns are built with abstract *anti-goals* and abstract *domain properties.* Abstract *anti-goals*, *domain properties* and predicates are reusable concepts defined on abstract domains that should be specialized in concrete domains at reuse time. **(Technique)** Hermoye et al. provide a formal reuse-based technique to reuse this library for threat analysis. They propose three main functions: *Retrieve* to get initial anti-goals, *Specialize* and *Adapt* to specialize each abstract variable (e.g., objects, agents, relations) of the attack pattern. **(Automation)** The KAOS method is supported by the Objectiver[2] tool. Even though we do not have details about technical aspects, the tool offers some functionalities such as modeling requirements and related concepts (goals, obstacles, expectations, hypotheses, etc.), querying the model to retrieve some model elements, exporting in XML format, and data exchanges in XMI format. Note that the tool does not handle the reuse of knowledge for SRE.

(Domain) The KAOS method considers the domain properties and expectations, which correspond to facts and assumptions in the conceptual framework. (Semmak, Gnaho, and Laleau, 2008) propose some extensions to the KAOS metamodel in order to express variability inherent to the Cycab transportation domain. The extensions consist mainly of the concept facet. Authors define the facet as a viewpoint or a dimension having an interest for domain (for example: the Cycab transportation domain is characterized by several facets like: "the localization mode", "the Road type"). Facets allow the structuration and organization of domain knowledge for reusability. Unfortunately, this extension was concerned with requirements elicitation in the Cycab transportation domain, but did not address security requirements elicitation.

- *Secure Tropos:*
  (Description) Secure Tropos method is derived from Tropos. The latter is a software

[2] http://www.objectiver.com/

development method based on the paradigm of agent-oriented software development (Bresciani et al., 2004)(Susi et al., 2005). Tropos deals with analysis, design, and implementation activities in a software development process, with a strong focus on the early phases of software development.

Mouratidis et al. (Mouratidis, 2006)(Mouratidis and Giorgini, 2007) extend Tropos with new concepts to cover security modeling (security constraints, secure dependencies, secure entities) and more. Secure Tropos distinguishes four main development phases: early requirements, late requirements, architectural design, detailed design, and architectural design. Recently, Secure Tropos was extended to be used in the field of cloud computing (Pavlidis et al., 2013)(Mouratidis et al., 2013)(Paja et al., 2012). **(Knowledge reliance)** In a previous work, Tropos method was extended with security patterns (Mouratidis, Weiss, and Giorgini, 2006). **(Form of representation)** Authors describe a pattern language, based on agent-oriented concepts. They used the Alexandrian format (Alexander, Ishikawa, and Silverstein, 1977) for organizing each pattern. In this format, the sections of a pattern are *context*, *problem and forces*, *solution*, and *rationale*. **(Reusable knowledge)** Authors proposed four main patterns: "Agency Guard" concerned with ensuring that there is only a single point of access to the agency to protect it from malicious agents. "Agent Authenticator" related to authentication of agency's agents. "Sandbox" related to mechanisms for separating running activities. "Access Controller" suggests intercepting all requests for the agency's resources. (**Technique)** Mouratidis et al. provide some guidelines and show how these patterns can be integrated within the architectural design stage of the Tropos agent-oriented methodology. **(Automation)** ST-Tool is one of the main tools known for Secure Tropos. Formal analysis is based on logic programming. ST-Tool (Giorgini et al., 2005b) provides a graphical user interface (GUI) that allows designers editing Secure Tropos models as graphs where nodes are actors and services, and arcs are relationships. To the best of our knowledge, ST-Tool does not handle the development using patterns for elicitation.

(Domain) Secure Tropos is not a method to define security requirements for a particular domain.

Another variant of Secure Tropos is presented in section (4.4).

B. Patterns not models

The identified SRE that (re) use patterns are presented below:

- *Okubo et al.*

  (Description) (Okubo, Kaiya, and Yoshioka, 2011) propose a method for security impact and security requirements analyzes. There are two types of security impact described with more details in the paper: *horizontal impact* on artifacts in the same stage and *vertical impact* on artifacts in different stages. **(Knowledge reliance)** The method proposed by Okubo et al. consists of two techniques: an analysis method of horizontal impacts using an extended misuse case; a combination of new security patterns and a traditional traceability technique to analyze security vertical impacts. The security patterns bridge the gap between security requirements and the design, so as to know impacts on code when security requirements change. **(Form of representation)** Okubo et al. constructed Security Requirements Patterns (SRPs) and Security Design Patterns (SDPs). A security requirement pattern is formed around a "context", a "problem", a "solution", and a "structure". In addition, a security design pattern has: "consequences", "implementation" and "sample code". **(Reusable knowledge)** In terms of knowledge, SRPs provide assets, threats. SDPs provide countermeasures. **(Technique)** The authors propose a process for security impact analysis that starts with selecting the SRP, identifying new assets, identifying new threats, identifying countermeasures, and finally, selecting the SDP, and estimate the impact for each countermeasure. **(Automation)** The method is not tool supported.

  (Domain) The method proposed by (Okubo, Kaiya, and Yoshioka, 2011) does not produce domain specific security requirements.

4. 2. METHODS THAT (RE) USE TAXONOMIES OR ONTOLOGIES

An ontology is a formal representation of the entities and relationships which exist in some domain. A taxonomy is an ontology in the form of a hierarchy. Whereas ontologies can have any type of relationship between categories, in a taxonomy there can only be generalization hierarchies. The SMAP revealed a variety of SRE methods that suggest the use of ontologies or taxonomies during a SRE process:

- *GBRAM*:

(Description) The objective of the Goal-based Requirements Analysis Method (GBRAM) (Antón and Earp, 2000) is to use goal and scenario-driven requirements engineering methods to formulate privacy and security policies, as well as requirements for e-commerce systems.

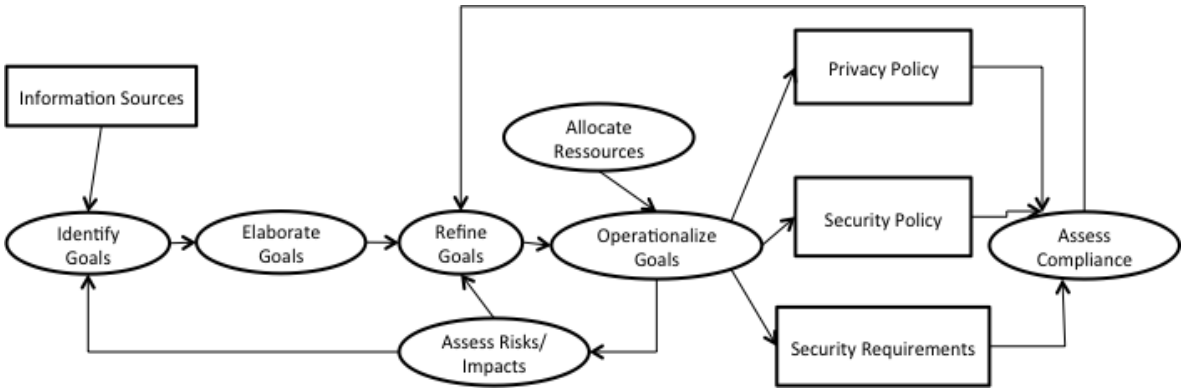


Figure 2.5. The GBRAM for Policy Formulation, taken from (Antón & Earp, 2000)

Furthermore, the method targets change management in organizational privacy and security policies, and system requirements (Figure 2.5). Lastly, the method is used to assure compliance of these system requirements to the privacy and security policies. In a later work built upon GBRAM, He and Antón (He and Antón, 2003) introduce a role-engineering framework called "Framework for Modeling Privacy Requirements in Role Engineering". In this framework, goals and scenarios are adopted in order to analyze permissions and establish role hierarchies, which then can be used to define a role-based access control model (RBAC).

Further, the authors suggest a context-free grammar for formalizing privacy goals articulated in natural language (Breaux and Antón, 2005). The formalized goals are used to analyze and compare the system goals stated through them. GBRAM is useful for analyzing and elaborating organizational goals—which are already integrated into policies—to elicit system requirements. By emphasizing and integrating the management of changes in the technology and the business environment to their method, the authors manage to include important aspects that many other methods ignore. Antón et al. (Antón and Earp, 2000) suggest using GBRAM at the beginning of the design phase in order to achieve the security of sensitive data. The heuristics are used to identify new, as well as previously overlooked, goals based on the results of risk assessment activities. The method is asset-centered and builds on the PFIRES approach for assessing risk in e-commerce systems (Antón and Earp, 2000).

(Knowledge reliance & Form of representation) Antón et al. (Antón and Earp, 2004) propose a requirements taxonomy for reducing Web site privacy vulnerabilities. They evaluated 25 Internet privacy policies from 8 non-regulated e-commerce industries. The evaluation permitted us to identify main goals and vulnerabilities. (**Reusable knowledge)** The security knowledge in the taxonomy was categorized into *Privacy Protection Goals*

and *Privacy Vulnerabilities.*

- *Privacy protection goals* express the desired protection of consumer privacy rights. They were categorized into five categories: *notice/awareness, choice/consent, access/participation, integrity/security* and *enforcement/redress.* For instance, *notice/awareness* goals assert that consumers should be notified or made aware of an organization's information practices before any information is actually collected from them. More details about the other goals can be found in Antón et al.'s publication (Antón and Earp, 2004).
- *Privacy vulnerabilities* reflect ways in which a Web site may violate consumer privacy. The seven main categories of privacy vulnerabilities are: *information monitoring, information aggregation, information storage, information transfer, information collection, information personalization,* and *contact.*

(Technique) The authors mentioned that Web site designers can use this taxonomy to ensure that their stated and actual policies are consistent with each other and it can be used by customers to evaluate and understand policies and their limitations. However, there were no precise procedures or techniques for the use. **(Automation)** As far as we could determine, the GBRAM method is not tool supported.

(Domain) GBRAM was designed for building secure e-commerce systems; the domain was described briefly and informally (Antón and Earp, 2000).

- *Secure Tropos*:

Another extension of the Tropos methodology was the one proposed by Massacci et al. (Mouratidis, Giorgini, and Manson 2003a)(Giorgini et al., 2006)(Giorgini et al., 2005a) (Massacci, Prest, and Zannone, 2004)(Massacci and Zannone, 2008). The authors use the Secure i* (Si*) language. In addition to the notions originally supported by the i* modeling framework, Si* introduces the notions of *delegation* and *trust*. *Delegation* is defined as a relation between two actors (the *delegator* and the *delegatee* — the one to whom something is delegated) and a goal, task, or resource (the *delegatum*). The notion of *trust* is used to separate delegation between trusted and untrusted actors. Similarly to delegation, trust is defined as a relation between two actors (the *trustor* and the *trustee*) and a goal, task, or resource (the *trustum*). A third extension to Tropos was proposed by Asnar et al. (Asnar et al., 2007)(Asnar, Giorgini, and Mylopoulos, 2006) for risk modeling; the Tropos Goal-Risk Framework, to assess risk based on trust relations among

actors. **(Knowledge reliance)** Massacci et al. (Massacci, Mylopoulos and Zannone, 2007) propose a formal ontology for socio-technical systems. **(Form of representation)** Authors formalized the concepts of Si* into an ontology. **(Reusable knowledge)** The concepts are organized into *extensional* and *intentional* predicates. *Extensional predicates* correspond to the edges and circles drawn by the requirements engineer (e.g., service, goal, task, resource, etc.) during the modeling phase. These predicates are used to formalize the intuitive description of the system. *Intentional predicates* are determined with the help of rules by the reasoning system; examples of these predicates are: aim(Actor:x,Service:s) and has_perm(Actor:x,Service:s). **(Technique)** The authors provide some axioms that define the semantics underlying Si*. They are used to complete the extensional description of the system. All these primitives were used to deal with the security organizational requirements.

The proposition (Pro) in the example below verifies whether an actor (X) who delegates the permission (perm) to another actor (Y) to deliver a service (S) is entitled to do it. With other prepositions, one can verify the authorization security requirement.

For example,

```
Authorization: Pro ← delegate(perm,X,Y,S) ∧ not has_perm(X,S)
```

(Automation) As far as we know, there was no automation support for this Secure Tropos extension. **(Domain)** This extension of Secure Tropos was not developed for a specific domain.

- *RITA :*

  (Description) Ivankina et al. (Ivankina et al., 2005), (Salinesi, Ivankina, and Angole, 2008) **(Knowledge reliance)** present a requirements elicitation method called RITA (Requirements Identification Threat Analysis) that makes use of a threat ontology. **(Form of representation)** Security requirements in RITA are expressed in forms of treatment that prevent threats. Treatments are formalized as goals. A goal is defined as "something that some stakeholder hopes to achieve in the future" (Rolland et al., 1998). A goal is expressed as a clause with a main verb and several parameters, where each parameter plays a different role with respect to the verb. Example of a security requirement in RITA (treatment): “Provide (connection help) $_{\text{object}}$ (to users) $_{\text{destination}}$ (when the connection fails) $_{\text{time}}$”.

(Reusable knowledge) The threats ontology in RITA organizes types of threats into classes and subclasses at several levels. Five classes are defined on the highest level: "User", "Design", "Environment", "Hardware" and "Engineering". Classes and subclasses are characterized by distinctive variables that help identify threats in the ontology, and define each class distinctively from the others. RITA also uses a second ontology that proposes a series of generic treatments for the generic threats identified in the threats ontology.

(Domain) To our knowledge, during the experimentations for evaluating RITA (Salinesi, Ivankina, and Angole, 2008), participants expressed the lack of domain specificity of requirements and threats.

- *Daramola et al.*

(Description) (Daramola et al, 2012a) (Daramola et al., 2012b) present an approach that leverages ontologies and requirements boilerplates in order to alleviate the effects of the lack of inexperienced personnel for Security Requirements Specification (SRS).

(Knowledge reliance) Daramola et al.'s approach makes use of ontologies and requirements boilerplates. **(Form of representation)** A requirement boilerplate (Hull, 2011) is a pre-defined structural template for writing requirement statements. The fixed parts of requirement boilerplate are reused when writing requirements, while the requirement engineer can manually fill in the parameter parts with information from its application.

An example of a boilerplate:

"BP2: The <system> shall be able to <action> <entity>"

The ontologies provide the necessary background knowledge required to identify security threats, and recommend appropriate countermeasures, while the requirements boilerplates provide a reusable template for writing Security Requirements in a consistent way in order to eliminate ambiguity. **(Reusable knowledge)** The Basic Threat Ontology (BTO) used in the approach contains a mapping of some kinds of security threats to specific defense actions based on information that was gathered from the literature and existing security ontologies. **(Technique)** The knowledge contained in the BTO is used for automatic recommendation of appropriate defense actions. This is made through ontology reasoning and other semantic capabilities.

(Automation) The proposed approach is tool-supported by the prototype ReqSec tool. ReqSec is an Eclipse-based tool that provides automated support for ontology-based security requirements specification by enabling the specification of security requirements from textual misuse case descriptions.

(Domain) To our knowledge, both the threats encapsulated in the BTO and the requirements boilerplates are generic and not domain specific.

- *Velasco et al.*

  (Description) (Velasco et al., 2009) propose an ontological representation for reusable requirements, which allows incompleteness and inconsistency in requirements to be detected and semantic processing in requirements analysis to be achieved. Note that the framework seems to be at an early stage, in the sense that it does not permit security requirements elicitation and analysis. To date, its contribution is limited to the proposed ontologies. **(Automation)** The framework is not supported with any tool **(Knowledge reliance & form of representation)** Velasco et al. defined some reusable knowledge encapsulated in ontologies. **(Reusable knowledge)** Authors defined two kinds of ontologies: a risk analysis ontology and a requirement ontology.

  - The risk analysis ontology is based on MAGERIT (MAGERIT, 2012), the information systems risk analysis and management method of the Spanish public administration. The ontology identifies five types of risk elements (*Asset*, *Threat*, *Safeguard*, *Valuation dimension*, *Valuation criteria*).
  - The concepts, meta-information and relationships included in the requirements ontology have been mostly taken from the authors' experience of requirements reuse-based method SIREN (Toval et al., 2001). The ontology organizes requirements into (*software requirements* and *system requirements*).

- *Salini et al.*

  (Description & Knowledge reliance) (Salini and Kanmani, 2012a) introduce a knowledge-oriented approach addressing the security requirements engineering phase for developing an E-Voting System. **(Form of representation)** For the knowledge part, the authors provided a security requirements ontology for e-voting systems. **(Reusable knowledge)** The terms used as ontology classes are the following: Stakeholder, Security Objective, Threat, Security Requirement, Asset, Vulnerability,

Security Requirements Pattern, Impact, Severity and Web application. The relations between the ontology classes and the properties used to represent the relations are uses, has, requires, is vulnerable to, implemented in, protects, mitigated by, provides, damages, affects, exploited by, addresses, assessed and part-of. Salini et al. explained that in practice, each security requirements pattern is matched with a set of Security Requirements during the ontology instantiation. A Security Requirements Pattern is defined as a set of Asset, Vulnerability, Threats and Impacts. In this way, one can start from the security objectives, find the Security Requirements Pattern that matches them and, thus, choose specific Security Requirements. Although the approach seems to be interesting and useful for defining security requirements, there was no validation reported for it, nor for the proposed security ontology. The ontology is still under development (not all identified security requirements have been mapped to the security objectives). **(Automation)** The approach is not supported with any tool.

(Domain) The approach was specifically proposed for security requirements engineering of e-voting applications.

- *Chikh et al.:*

  (Description & Knowledge reliance) (Chikh et al., 2011) present a framework for building security requirement specifications related to Information Security Requirements (ISRs) using ontologies. **(Form of representation)** The framework uses three kinds of generic ontologies as a solution to this problem – software requirement ontology, application domain ontology, information security ontology. However, despite the fact that the framework looks promising, it is difficult to know its usefulness, since no validation was presented; **(Automation)** the authors mentioned ongoing development of a prototype to evaluate their proposition.

  (Domain) The framework presented by Chikh et al. suggests the use of a domain application ontology. However, up to date, to the best of our knowledge no further publications describe this use in more details.

4.3. METHODS THAT (RE) USE TEMPLATES OR PROFILES

Some SRE methods rely on templates and profiles as another kind of reusable knowledge for SRE. The identified methods that (re) use this forms are:

- *Zuccato et al.:*

(Description) (Zuccato, Daniels, and Jampathom, 2011) present an approach that organizes security requirements engineering around five activities. The first activity starts with a simplified risk analysis approach by means of questionnaires to identify areas in the business which can have security problems. Subsequently, the security requirements for the development project are selected (requirement profiles). These requirements are then forwarded to the suppliers. **(Knowledge reliance & form of representation)** The method proposed by Zuccato et al. is based essentially on the use of security requirements profiles that address a business domain, in a commercial organization, where activities have to serve a business purpose (not to be confused with security patterns which describe a security domain solution according to authors). **(Reusable knowledge)** Typical examples for this business orientation would be IP-TV Services (e.g. renting a movie, recording some programs, delayed viewing, ...), VoIP (e.g. multiple numbers, location locking, answering machine, ...) or customer self-administration (e.g. myPages, myWorkingPages, MyFamily, Mobile Device Management (MDM)…) where a profile is created for the service category and then reused, with some adaptation, for the specific service. **(Automation)** The approach is not tool supported.

(Domain) Zuccato et al. use the term "service security requirement" and deal with security problems when developing and operating telecommunication services.

- *Firesmith:*

(Description, Knowledge reliance & Form of representation) (Firesmith, 2004) suggests using textual security requirements templates (not to be confused with security use cases templates). An example of a reusable parameterized template for specifying an integrity security requirement:

"The [application center/business unit] shall protect the data it transmits from corruption (e.g., unauthorized addition, modification, deletion or reply) due to [unsophisticated/ somewhat sophisticated/sophisticated] attack during execution of [a set of interactions/use cases] as indicated in [specified table]."

Users of these templates can manually replace the brackets according to their different applications.

(Reusable knowledge) More detailed templates in Firesmith's research could not be found. The proposition of the author mainly concentrates on arguing the importance of specifying the knowledge into this kind of templates. **(Technique)** The author

provides an asset-centered and reuse-based procedure for requirements and security teams to analyze security requirements containing 13 steps, starting by identifying the valuable assets, identifying threats, and estimating vulnerabilities. The steps end by specifying requirement through the instantiation of templates based on the parameters from the previous steps. **(Automation)** This proposition was not tool supported.

(Domain) The reusable templates proposed by the author are generic, not specific to a particular domain.

4.4. METHODS THAT (RE) USE CATALOGS OR GENERIC MODELS

Some SRE methods define generic models of common security problems and their solutions, in order to (re) use them. Some others rely on catalogs to encapsulate the reusable knowledge as presented below:

- *Misuse Cases:*

  (Description) Sindre & Opdahl (Sindre and Opdahl, 2001a)(Alexander, 2002)(Sindre and Opdahl, 2001b)(Sindre and Opdahl, 2005) extend the traditional use case approach to also consider misuse cases, which represent behavior not wanted in the system to be developed. Misuse cases are initiated by misusers. They have have two representations: a graphical diagram and a textual specification.

  (Knowledge reliance) Misuse cases were initially developed without relying on any kind of knowledge repositories. However, Sindre et al. (Sindre, Firesmith, and Opdahl, 2003) defined an approach based on a repository of generic misuse cases (generic threats and generic security requirements). **(Form of representation)** Sindre et al. represent the reused knowledge using generic misuse cases. Each misuse case has a name, summary, preconditions, misuser interactions, systems interactions and post-conditions.

  (Reusable knowledge) Authors suggest two main reusable artifacts: generic threats (e.g., spoofing, i.e., a misuser gaining access to the system by pretending to be a regular user) and generic security requirements (e.g., access control) described independently of particular application domains. **(Technique)** Authors provide a way to use/reuse this repository through some guidelines. **(Automation)** As far as we know, misuse cases are still not tool supported.

  (Domain) As far as we have been able to determine, misuse cases were not developed for special domains.

- *Abuse frames.*

(Description) Lin et al. (Lin et al., 2003b)(Lin et al., 2003a)(Lin et al., 2004) define so-called anti-requirements and the corresponding abuse frames. Their proposition is comparable to problem frames introduced by Jackson (Jackson, 2001). An anti-requirement specifies the undesirable phenomena in the system that must be prevented from happening; it expresses the intentions of malicious users. An abuse frame (Figure 2.6) represents a security threat. Authors incorporate anti-requirements into abuse frames to represent a security threat. The authors state that the purpose of anti-requirements and abuse frames is to analyze security threats and derive security requirements.

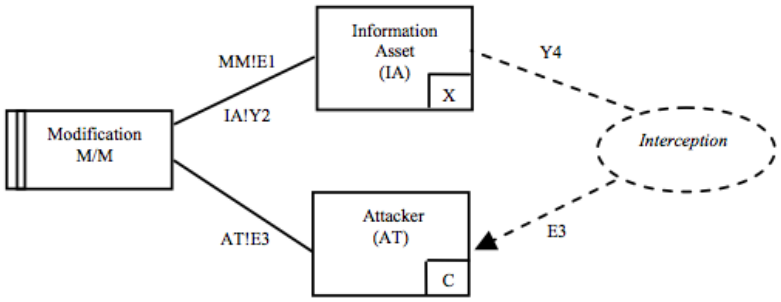


Figure 2.6. A standard modification abuse frame taken from (Lin et al, 2003)

(Reusable knowledge) In problem frames, each frame describes a particular problem class (e.g., Information Display, Workpiece, and Required Behavior frames). Similarly, Lin et al. propose abuse frames that describe classes of security violation (interception, modification, and denial of access). Each one represents a threat that can violate a particular security goal. **(Knowledge reliance & Form of representation)** These security violations represented through abuse frames diagrams are meant to be reusable. Figure 2.6 shows a standard modification frame. Modification arises whenever there is an information asset in the physical world that an attacker wishes to change. The problem is to find a modification machine that allows an attacker to achieve it. Modification violates integrity. **(Technique)** The authors propose an iterative threat analysis method that essentially comprises four steps (scoping the problem and identify the sub-problems, identifying the threats and constructing abuse frames, identifying security vulnerabilities, addressing security vulnerabilities). **(Automation)** As far as we know, abuse frames are not supported with a tool. **(Domain)** Abuse frames were not designed for a particular domain.

- *Security Use Cases:*

  (Description) Nearly at the same period when misuse cases appeared to the community, Donald Firesmith presented security use cases (Firesmith, 2003). He pointed out that misuse cases are effective ways of analyzing security threats but are inappropriate for the analysis and specification of security requirements. Misuse cases concentrate on interactions between the application and its misusers who seek to violate its security. While security use cases are meant to specify requirements that the application shall protect itself from its relevant security threats.

  (Knowledge reliance) Firesmith tried to keep the security use cases templates (Firesmith, 2003) at a reasonably high-level of abstraction for reusability purposes. **(Form of representation)** Security use cases have a name, a path, a security threat, preconditions, misuser interactions, system requirements and post-conditions. **(Reusable knowledge)** The author presented the reusable use cases: access control, integrity and privacy. **(Technique)** The author emphasizes that, when reused on real projects, each path template can be made more specific to the application by replacing the general words "system" and "user" with the specific application name and the specific type of user. **(Automation)** To our knowledge, security use cases are not tool supported.

  (Domain) Security use cases were not proposed for domain specific security requirements.

  - *Saeki and Kaiya:*

    (Description) (Saeki and Kaiya, 2009) propose a weaved security requirements elicitation method that uses **(Knowledge reliance)** Common Criteria (CC) (MD, 2002) and related knowledge sources to identify security requirements from functional requirements through eliciting threats and security objectives (Figure 2.7). **(Reusable knowledge)** The authors think that CC can be considered as a kind of catalog to provide knowledge on threats, security objectives and security functions that have generally appeared. For example, by using Common Criteria, one can select the objective "data encryption" from the catalog, to mitigate the threat "disclosure of password data". **(Technique)** The proposed technique is to weave through Common Criteria two types of requirements elicitation; one is any existing functional requirements elicitation, and the other is a typical method for eliciting security functional requirements. **(Form of representation)** The method relies on CC as a

source of knowledge to support activities of security requirements elicitation. The authors used CC Part 2 (CC, 2007), which has about 120 Security Functional (SF) components, as a catalog. In addition, they used ECMA- 271 E-COFC (ECMA, 1999) (which can be considered as a profile of CC in a certain problem domain), as catalogs of threats and security objectives. As shown in the right hand side of Figure 2.7, the method accumulates a threat catalog; a security objective catalog and a SF component catalog, and holds relationships between their catalog entries (i.e. security objective mitigates threat, SF component represents security objective). **(Automation)** As far as we know, the proposition of Saeki and Kaiya is not tool supported. **(Domain)** The proposed method is not devoted to a particular application domain.

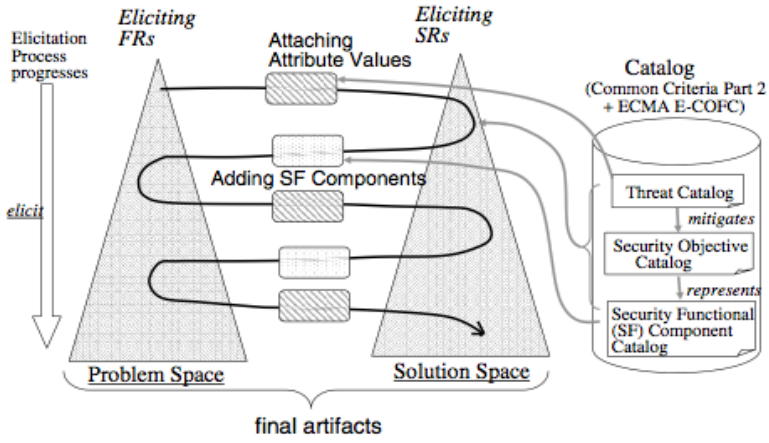


Figure 2.7. Using knowledge included in Common Criteria, taken from (Saeki and Kaiya, 2009)

- *SIREN for security requirements* :

(Description) Toval et al. (Toval et al., 2001) propose an approach for security requirements elicitation. **(Knowledge reliance)** The approach is a particularization of SIREN (SImple REuse of software requiremeNts), a general-purpose RE method based on requirements reuse. The particularization of SIREN to the security profile has been based on the risk analysis and management method MAGERIT (MAGERIT, 2012). Security requirements specify the countermeasures prescribed by MAGERIT after the risk analysis. Therefore, it is the MAGERIT risk analysis and management that determines the security mechanism to be used in each circumstance. SIREN encompasses a process model and some guidelines. **(Form of representation)** The guidelines that SIREN provides consist of a *hierarchy of requirements specification documents* together with the templates for each document. These serve to structure a

reusable requirements repository. **(Domain & Reusable knowledge)** The repository defined in SIREN contains functional and non-functional requirements from specific domains and profiles. A SIREN profile consists of a homogeneous set of requirements that can be applied to a variety of domains, such as information systems security, and the personal data privacy law. There are two main types of requirements in the repository:

- Parameterized: this kind of requirements contains some parts that have to be adapted to the application being developed at the time. If this requirement is chosen, the parameterized part will be instantiated, that is, the information in brackets will be replaced by a specific value according to the current project. For example: "SRS.3.5.3.1.S.301 The security manager shall check the user's identifiers every [time in months] to detect which ones have not been used in the last [time in months]."
- Non-parameterized: requirements that could be applied directly to any project concerning the profiles and/or domains in the repository. For example: "SRS.3.4.3.S.5. The firewall configuration will be screened host."

(Technique) Toval et al. adapted a spiral life cycle in SIREN to take requirements reuse explicitly into account in the RE process. Details about the process can be found in (Toval et al., 2001). **(Automation)** To our knowledge, SIREN is not tool supported.

- *Secure Tropos:*

(Knowledge reliance) In their recent work (Mouratidis et al., 2013)(Pavlidis et al., 2013), Mouratidis et al. suggest considering the activity of cataloging during the elicitation and analysis process. The main aim of this activity is to develop a reference catalog model that can be employed not just for the project for which it was initially developed but can work as a reference model for any projects that demonstrate similar characteristics. **(Form of representation)** The reference catalog diagram takes the form of a reference model that contains graphical representation of different concepts needed for elicitation process. **(Reusable knowledge)** The reference catalog provides relationships between the concepts security and privacy goals, threats, security and privacy measures, and security and privacy mechanisms. For example, the security goal "availability" can be threatened by the threats "Data Location" and "Insecure Storage". The measure to mitigate these threats can be "API Interoperability" which

uses the mechanisms "Middleware, Support Multiple Providers". For their case study, authors used existing information in the security document of the company to construct a cataloguing diagram. **(Automation)** The framework is supported by a tool, which has been developed based on the Open Models Initiative ADOxx Platform. The tool provides an environment for developers to create a number of diagrams that support the process of the method. In particular, the tool permits development of the Security and Privacy Reference Catalog Diagram discussed before.

4. 5. METHODS THAT (RE) USE MIXED FORMS OF SECURITY KNOWLEDGE

There are a variety of SRE methods that (re) use different (mixed) forms of knowledge; our SMA identified the following ones:

- *SQUARE:*

  (Description) SQUARE (Mead and Stehney, 2005)(Mead et al., 2008)(Mead and Hough, 2006) is a comprehensive methodology for security requirements engineering. Its aim is to integrate security requirements engineering into software development processes (Mead et al., 2008). SQUARE stresses applicability in real software development projects and thus provides an organizational framework for carrying out security requirements engineering activities.

  (Knowledge reliance) Travis et al. introduced a new variant of SQUARE; R-SQUARE (Christian, 2010) which is defined using SQUARE Lite as a base model and incorporating reuse in some places of the process. **(Reusable knowledge)** However the introduced layer of reusable knowledge gives only some indications and no more. Throughout the selected publications, it was not possible to access to this reusable knowledge. For example, during the "agree on definitions" step, the authors suggest creating and maintaining a glossary of relevant terms and definitions so that the meanings of requirements do not become ambiguous over time as they are reused. During the identification of assets and goals step, the authors recall that organizations that develop product lines of secure software (Mellado, Fernandez-Medina, and Piattini, 2008) will likely have overarching business and security-related goals that are intended to apply to all affected projects. During the risk assessment phase, R-SQUARE method suggests to use threat models, which are known to be abstract and highly reusable. **(Automation)** SQUARE has been automated by means of the P-

SQUARE tool; this tool is designed for use by stakeholders, requirements engineers, and administrators. It supports both the security and privacy aspects of SQUARE by recording definitions and searching and adding new terms, identifying the project business goals, assets, and security or privacy goals, adding or editing links to project artifacts performing risk assessment and identify threats. No technical details were provided concerning the tool. The tool P-SQUARE does not support R-SQUARE (Reusable SQUARE). **(Domain)** The SQUARE method is not intended to elucidate and analyze the security requirements of a particular domain.

- *SREP :*

  (Description) Mellado et al. (Mellado, Fernández-Medina, and Piattini, 2006) (Mellado, Fernández-Medina, and Piattini, 2007) present the Security Requirements Engineering Process (SREP). SREP is an iterative and incremental security requirements engineering process, which is based on the Unified Process (Jacobson, Booch, and Rumbaugh, 1999) software life-cycle model with multiple phases. **(Knowledge reliance)** SREP is asset based, risk driven, and, following the Common Criteria (CC) (Mellado, Fernández-Medina, and Piattini, 2007) supports the reuse of security requirements, as well as the reuse of knowledge on assets, threats, and countermeasures. **(Form of representation)** It relies on a Security Resources Repository (SRR), which stores some reusable security elements that are of different forms: plain text, security use cases, attack trees, misuse cases. **(Reusable knowledge)** The meta-model representing the organization of the SRR is exposed in Figure 2.8. The most important aspects of it are:

  - Generic Threat and Generic Security Requirements are described independently of particular domains.
  - Security Requirement Cluster is a set of requirements that work together in satisfying the same security objective and mitigating the same threat.
  - The Req-Req relationship allows an inclusive or exclusive trace between requirements. An exclusive trace between requirements means that they are mutually alternative, as for example that they are in conflict or overlapping. Whereas an inclusive trace between requirements means that to satisfy one, another (others) is (are) needed to be satisfied.

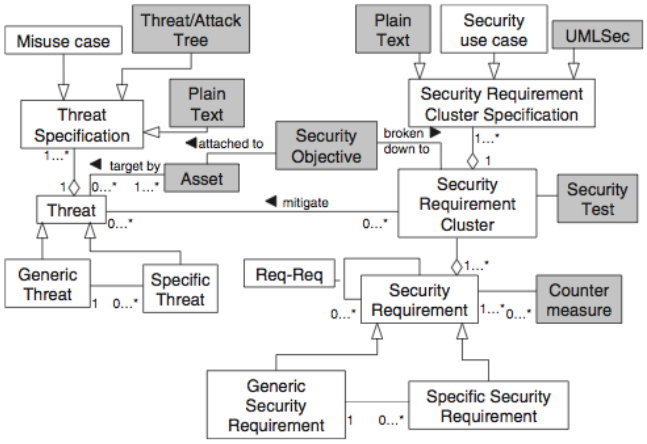


Figure 2.8. Metamodel for security resources repository taken from (Mellado, Fernández-Medina, and Piattini, 2007)

(Automation) Tool support is critical for the practical application of the SREP in large-scale software systems due to the number of handled artifacts and the several iterations that have to be carried out. However the authors have not developed it so far.

(Domain) The SREP is not designed to define security requirements of a specific domain. Although the authors mentioned briefly that SRE Process uses public-domain sources and threat lists during the assets/threat analysis phase.

4.6. METHODS THAT DO NOT (RE) USE SECURITY KNOWLEDGE

The SMAP found that there are a wide variety of SRE methods that do not consider knowledge reuse during the SRE process; we summarize them below. Note that for this category, we obviously skip answering the questions related to reusable knowledge, form of representation, and technique. Since there is no knowledge reliance, we cannot talk about the points related to the knowledge reused, its form of presentation, or the technique for reusing it.

- *Secure I*:*
  (Description) Liu et al. (Liu, Yu, and Mylopoulos, 2002)(Yu and Liu, 2001)(Liu, Yu, and Mylopoulos, 2003) propose employing explicit modeling of relationships among strategic actors in order to elicit and analyze security requirements. Authors analyze attackers, vulnerabilities in actors' dependency network, countermeasure, and access control. In this contribution, all actors are assumed potential attacker, which inherit capabilities, intentions, and social relationships of the corresponding legitimate actor. The basic idea of dependency analysis is that dependency relationships bring vulnerabilities to the

system and the depending actor. The dependency vulnerability analysis aims to find which dependency relationship is vulnerable to attacks. In this regard, the actors, in the basic dependency model, are substituted with its corresponding attacker, and then the impact of the attack to the dependency relationship is traced to the network of actors.

(Knowledge reliance) Authors mentioned that it would be useful to retrieve attacks and prototypical solutions from pre-defined taxonomies or knowledge repositories (Liu, Yu, and Mylopoulos, 2002), but the method, as it is, does not handle the use of this kind of knowledge so far. **(Automation)** Secure I* was not initially tool-supported (Liu, Yu, and Mylopoulos, 2002). Later, Giorgini et al. adapted Secure I* concepts within Secure Tropos and proposed ST-tool (Giorgini et al., 2005b) which is used today for secure I* diagrams. As far as we know, the tool does not support reuse of knowledge for SRE.

(Domain) Secure I* was not designed to be used in a particular domain.

- *UMLSec* and *SecureUML:*

  (Description) UMLSec and SecureUML are two main UML-based extensions for modeling security. SecureUML (Lodderstedt, Basin, and Doser, 2002)(Araujo and Gupta, 2005) is a UML-based modeling language for the model-driven development of secure systems (Lodderstedt, Basin, and Doser, 2002). SecureUML takes advantage of Role-Based Access Control (RBAC) for specifying authorization constraints by defining a vocabulary for annotating UML-based models with information relevant to access control. Using UMLSec (Jürjens, 2002)(Jürjens, 2005)(Jürjens and Shabalin, 2004)(Best, Jurjens, and Nuseibeh, 2007), security requirements are defined by assigning security stereotypes, constraints, and tagged values, which are defined in a UML profile for the elements of the design models. **(Knowledge reliance)** Neither UMLSec nor SecureUML considers the (re) use of security requirements knowledge.

  (Automation – SecureUML) Araujo et al. (Araujo and Gupta, 2005) presents a SecureUML template — a Microsoft Visio template built to model authorization systems. The tool allows architects to model their role-based access control systems. We could not find technical information about the SecureUML template. According to Araujo et al., the proposed template assists developers by identifying poor authorization design and implementations, helping to find contradictions/holes such as backdoors, or identifying authorization bypass opportunities.

(Automation - UMLSec) Jürjens et al. (Jürjens and Shabalin, 2004) presents a framework for verification of UMLSec models for security requirements. The framework provides three input and output interfaces for the analysis plug-ins: a textual command-line interface, a graphical user interface, and a web-interface. Inputs can be UML diagrams in the form of XMI files, as well as textual parameters. As output can be UML diagrams as XMI (or .zuml) files and text messages. Advanced users of the UMLSec approach can use the tool to implement verification routines for the constraints associated to self-defined stereotypes. A new UMLSec implementation variant called CARiSMA (Wenzel, Warzecha, and Jürjens, 2012) has been existing since 2012, and this time is based on the Eclipse Modeling Framework. CARiSMA enables users to perform compliance, risk, and security analyses. **(Reusable knowledge)** Neither the automation for UMLSec nor for SecureUML supports the reuse of knowledge. **(Domain)** Neither UMLSec nor SecureUML define security requirements for specific domains.

- *CORAS* :

(Description) Dahl et al. (Dahl, Hogganvik, and Stølen, 2007)(Lund, Solhaug, and Stølen, 2011)(Vraalsen et al., 2005) (Hogganvik and Stølen, 2006) present an organizational model-based method that covers threat, vulnerability, and security risk analysis. It also covers the elicitation of security goals. The language consists of five different kinds of diagrams: *asset diagrams*, *threat diagrams*, *risk diagrams*, *treatment diagrams*, and *treatment overview* diagrams. Their basic building blocks are presented in Figure 2.9.

Figure 2.9. Basic building blocks of the CORAS diagrams taken from (Dahl, Hogganvik, and Stølen, 2007)

(Knowledge reliance) We could not find any papers that present the CORAS method (re) using security knowledge. **(Automation)** The CORAS Tool (Vraalsen et al., 2005) follows a client-server model and is developed entirely in Java. The CORAS client

application permits the analyst to create new analysis projects and documents, to edit security analysis results, to generate analysis reports, and to manage and reuse experiences from previous analyses. The latest version (Lund, Solhaug, and Stølen, 2011) has in addition a user interface containing a pull down menu, a tool bar, and a palette that contains all model elements. The CORAS tool does not support the reuse of knowledge.

(Domain) As far as we know, CORAS is not a domain specific SRE method.

- *ISSRM*:

(Description) Mayer et al. (Mayer, 2012) propose a risk-based security requirements engineering framework that focuses on integrating risk analysis with requirements engineering activities. The main idea is to align Information Technology (IT) security with business goals. For this aim, the impacts of risks on business assets are analyzed; threats and vulnerabilities in the architecture are identified, and security requirements are defined, to mitigate the risks.

(Knowledge reliance) ISSRM approach does not rely on any kind of knowledge repositories. **(Automation)** ISSRM approach is not tool supported.

(Domain) ISSRM was not dedicated to a particular domain.

- *Morda:*

(Description) Evans et al. (Evans et al., 2004)(Buckshaw et al., 2005) propose Mission Oriented Risk and Design Analysis (Morda) as a methodology for analyzing security risks. Morda combines threats, attacks, and mission impact concepts for deriving an unbiased risk metric. **(Knowledge reliance)** Through this literature review, no publication addressing security knowledge (re) use by Morda was found. **(Automation)** Morda is not supported by a tool.

(Domain) Through the literature, we did not find any use of security knowledge by Morda. The approach is not for a particular domain, and does not consider domain knowledge.

- *CRAC++:*

(Description) Morali and Wieringa present a method named CRAC++ (Morali and Wieringa, 2010), which is an extension of the older method CRAC (Morali et al., 2009) The Confidentiality Risk Assessment and Comparison (CRAC) is an architecture-based

method for confidentiality risk assessment in IT outsourcing. In CRAC++, the method is extended with a step to identify confidentiality requirements in outsourcing. In other words, the method specifies and identifies confidentiality requirements of the client that are not implied by the known confidentiality requirements of the provider, and which therefore are candidates for inclusion in a Service Level Agreement (SLA) with that provider. Authors present a case study to evaluate the method. **(Knowledge reliance)** To the best of our knowledge, CRAC++ does not rely or use predefined reusable structured knowledge. **(Automation)** CRAC++ is not equipped with a tool.

(Domain) To the best of our knowledge, CRAC++ does not define security requirements for specific domains.

- *SREF* :

  (Description) Haley et al. (Haley et al., 2008)(Haley et al., 2006) present a framework for security requirements elicitation and analysis called SREF (Security Requirements Engineering Framework). The framework considers security in an application context; it represents security requirements as constraints, and develops satisfaction arguments for the security requirements. The system context is described using a problem-oriented notation, then is validated against the security requirements through construction of satisfaction arguments.

  (Knowledge reliance) To the best of our knowledge, Haley et al. did not rely on knowledge reuse for their proposed SREF. **(Automation)** No tool is presented with this framework.

  (Domain) The SREF method was not designed for a particular domain, though it was applied in the aircraft domain for evaluation. The authors (Haley et al., 2008) report needing the help of the project's domain experts at the step concerned with determining the assets, the harm that the assets can suffer from, and the security goals to avoid those harms. On the other hand, the SREF method leaves to the analyst the use of trust assumptions when deciding whether an element of a domain is relevant to consider during the security analysis.

- *MSRA:*

  (Description) Multilateral Security Requirements Analysis method (MSRA) (Gürses, Berendt, and Santen, 2006)(Gürses and Santen, 2006), aims to apply the principles of multilateral security (Rannenberg, 1993) during the requirements

engineering phase of systems development. This is done by i) analyzing security and privacy goals of all the stakeholders of the system-to-be, ii) identifying conflicts, and iii) consolidating the different stakeholders' views. The method proposes analyzing conflicts carefully and solving them either during requirements analysis, through design, or using negotiation mechanisms at runtime. It borrows both from theories on multilateral security and viewpoint-oriented requirements engineering. Security requirements are then derived from the reconciliation of multilateral security goals.

(Knowledge reliance) The method does not rely on reusable knowledge for security requirements elicitation. **(Automation)** No tool is presented with the method, to the best of our knowledge.

(Domain) The authors give informal textual definitions of security and privacy related to ubiquitous technologies domain. They report some kinds of privacy threats possible through these new technologies. Then, they present the MSRA method that deals with the complexities of articulating privacy and security requirements for ubiquitous systems.

5 SUMMARY

The systematic mapping study recalls a great interest in security requirements engineering with a considerable attention to the (re) use of knowledge for defining security requirements. This section returns to the main research questions of this systematic mapping study and replies to them according to all the publications retrieved.

The following summarizes the answers to the research questions.

RQ1. Does the security requirements method rely on reusable knowledge?

Our results indicate that reuse knowledge is addressed in 29 (31%) out of 95 papers. This allows us conclude that overall, the deployment of reusable knowledge in security methods is relatively unexploited and possibly immature. The rate of evaluation papers found (only 10.5%) indicates that most of the propositions are not evaluated regarding their applicability and usability in large-scale case studies and with end users from practice. One might say that this is due to the fact that most of these methods were proposed in an academic context, mostly through PhD dissertations focusing on validating the proposition in a small-scale laboratory experiment rather than in large-scale case studies. Nevertheless, this indicates that

more attention should be given to the applicability and usability of the deployment of knowledge (re) use in SRE.

RQ2. How is the reused knowledge represented?

Surveying the different proposals allowed us to identify different forms of knowledge representation: Patterns constituted 9.4% of them, taxonomies and ontologies 13.6%, templates and profiles 2.1%, catalogs and generic models 10.5%. Few propositions (3.1%) used a mix of these different forms. The rest of the proportion concerns proposals that don't reuse security knowledge. This gives us a picture about the different forms to represent and access knowledge proposed in the literature. The question remains of why some representations are more "popular" than others, and it would be interesting to find out more directly from academics and practitioners (through a survey) about why they may prefer some forms to others. For instance, one might suggest the hypothesis that ontologies are known to feature reasoning mechanisms, catalogs might be easy to access and generic models might be easy to visualize for re-use.

In any case, the following summarizes the different forms of knowledge representation identified:

- **Security patterns:**
    - **Models:** Let us mention notably the work of Hermoye et al. (Hermoye, van Lamsweerde, and Perry, 2006) who propose an attack pattern library containing attack trees using the KAOS framework, and the proposition of Mouratidis et al. (Mouratidis, Weiss, and Giorgini, 2006) who enforce the Secure Tropos method with security patterns models.
    - **Not models:** The method proposed by Okubo et al. (Okubo, Kaiya, and Yoshioka, 2011) makes use of security requirements patterns and security design patterns.

- Generic models: Some researches propose repositories of generic models for the purpose of reuse, such as generic misuse cases (Sindre, Firesmith, and Opdahl, 2003), security use cases (Firesmith, 2003), abuse frames diagrams (Lin et al., 2004).

- Security requirements templates (plain text): Firesmith suggests reusable security requirements templates (Firesmith, 2004). SIREN relies on a repository of parameterized and non-parameterized security requirements (Toval et al., 2001).

- **Ontologies:** Some approaches propose the use of ontologies for SRE (Daramola et al., 2012a)(Salinesi, Ivankina, and Angole, 2008)(Dritsas et al., 2006)(Salini and Kanmani, 2012a)(Velasco et al., 2009)(Chikh et al., 2011); most of them are in their early stages and not yet validated. In fact, none of them are used by existing categories of security requirements.

- **Taxonomies:** As a continuity of the proposed method GBRAM, Antón and Erap propose a taxonomy for reducing web sites privacy vulnerabilities (Antón and Earp, 2004).

- **Catalogs:** The recent work of (Mouratidis et al., 2013) suggested relying on a catalog of reusable models, but did not mention what these models contain exactly and how to use them. Saeki and Kaiya's (Saeki and Kaiya, 2009) method makes use of Common Criteria catalogs that contain threats, security objectives and Security Functional (SF) components.

- **Profiles:** Zuccato et al.'s method uses what the authors call security requirements profiles (Zuccato, Daniels, and Jampathom, 2011).

- **Mixed:** The method SREP (Mellado, Fernández-Medina, and Piattini, 2006) relies on a Security Resource Repository (SRR) which stores reusable security elements that can be represented in different forms (misuse cases, attack trees, security use cases, UMLSec, plain text). The method R-Square (Christian, 2010) also uses different kinds of reuse structures (definitions, glossaries, threat models).

RQ3. What are the techniques for (re) using the knowledge?

Most of the approaches (14.7%) provide manual guidelines for reuse; some of them add a process to follow. Few rely on semi-automated techniques (10.5%) such as formal rules. The ontology-based approaches take advantages of reasoning features of ontologies. These results indicate a high tendency to re-use through manual guidelines and a low trend to automatic techniques (only 5%), which can be seen as a weakness. By that, we mean that starting with a well-formalized knowledge source then re-using it through a human activity following some guidelines may lead to negative results if the process is not applied well.

RQ4. What are the main reused elements?

The main reused elements are often threats (26.3%) and security requirements (30.5%). The reused knowledge might differ slightly from one approach to another, but there is always knowledge related to the dark side of security (threats) and the treatment side (security requirements). Very few approaches reuse knowledge related to the organization and its

assets (5.2%). So what about the organizational side where threats appear and arise? (The assets to protect and their locations, the different persons involved in an organization, the organizational activities…) This knowledge can be reused too through different projects. In addition to threats, there are the attackers, or categories of attackers, their attack methods and their attack tools, classes of vulnerabilities and common impacts of threats. Research on re-use of knowledge in SRE should consider more elements of security and not just requirements and threats.

RQ5. Are they tool supported?

Among the 95 selected papers, only 13.6% propose tool support. However, most approaches do not provide tools that handle the *reuse of knowledge*, except one approach (Daramola, Sindre, and Stalhane 2012), where the authors present only a prototype. The tool mentioned by Mouratidis et al. (Mouratidis et al., 2013) provides a way to create the reference catalogue diagram and reuses it as discussed before. This indicates that most propositions are unfortunately not tool supported. A possible explanation can be, as stated for RQ1, namely that in the academic environment where these methods were proposed, tool implementation is not the main focus.

RQ6. Are the security requirements specified for a specific domain? For which domain? How they are represented?

Only 7 % of selected papers addressed security requirements for specific domains. In all the other cases, the propositions remain generic; it is left to the requirements engineer and the security analysis to adapt these propositions to the domain at hand. The spotted domains were e-commerce systems, e-health applications, e-voting applications, and ubiquitous technologies. These are sensitive domains but securities of many other domains need to be considered. The other point noticed is, even when the domain is used, it is often described textually and informally, not necessarily with reusable elements. Table 2.3 (in page 90) summarizes the domain related questions in SRE.

RQ7. What are the new challenges regarding security knowledge (re) use in SRE?

Based on the SMAP presented in this chapter, the challenges in the following are part of the authors' own view of open questions:

(Challenge 1) It is interesting to note that the risk-based approaches found do not handle reuse of security knowledge. The challenge will be to reconsider knowledge reuse in these

methods. (Challenge 2) Many approaches for SRE relying on ontologies are emerging. They seem to be at their early stages and have not been validated yet. The challenge is to strengthen them and to apply them to large-scale case studies. (Challenge 3) Ontology-based approaches are not handled by the existing security requirements engineering categories (model based); it would be interesting to see how to merge these two directions. (Challenge 4) The domains for which security requirements are defined need be taken in consideration during the SRE process (Challenge 5) There is a lack in automated support that handles knowledge reuse for the different SRE methods. More tools to support that would be appreciated. (Challenge 6) It would be interesting to generalize and unify all these efforts (like in UML), so that they can more easily be exploited by industrials.

Table 2.4 (in page 91) summarizes the results of the systematic mapping study. The columns contain the different SRE methods grouped into categories. The lines contain the main aspects of knowledge reuse (form of representation, reusable knowledge elements, reuse technique and automation). The code used to fill the cells of the matrix is also presented. Cells marked with "-" mean that the method does not take in consideration the corresponding aspect of knowledge reuse.

6 RELATED WORKS

To the best of our knowledge, no research exists in the literature to review in a systematic way the issue of knowledge reuse in security requirements engineering. One worth mentioning work though is that of Chernak (Chernak, 2012) who conducted an online survey on requirements reuse in 2010. His survey reports that 80% of participants find that reuse is important and brings benefits. Yoshioka et al. (Yoshioka, Washizaki, and Maruyama, 2008) presented a survey limited to security patterns. This is interesting, but the other forms of reuse are neglected, whereas they were taken in consideration in this chapter.

Devanbu et al. (Devanbu and Stubblebine, 2000) is one of the older references that presented a "brief" survey on security models and requirements. Recently, some publications were dedicated to security requirements engineering (Fabian et al., 2010),(Mellado et al., 2010), (Salini and Kanmani, 2012b), (Tondel, Jaatun, and, Meland 2008). However, none of these existing reviews tackled the specific issue of "knowledge reuse".

7 THREATS TO VALIDITY

Like with empirical researches, there are threats to the validity. In the following some threats that might compromise our results are cited:

Search engines used in the SMAP (External validity)

All retrieved results rely on the functionality and precision of the search engines of the used digital libraries. Unfortunately, many search engines of computer science digital libraries turned out to be unreliable. Moreover, the results were also based on digital libraries for which our institution has subscription to. Unfortunately, it was not possible to explore a system like SCOPUS, which is known to be particularly useful because it indexes publications from a large number of publishers.

Selected sources (Construction validity)

In this research, the SMAP was more focused on publications' sources related to the security requirements engineering field than on those related to the knowledge engineering field. This makes the results subject to discussion and comparison with other SMAPs' results that might address the subject in the other way around, "security requirements in knowledge engineering" for example. Moreover, being researchers in the area of requirements engineering and information systems, there is a risk that we may have been biased by our experience and collaborations in the selection and the analysis despite our effort to avoid it. For example, some selected studies of the mapping were proposed by colleagues from the same lab colleagues. In particular, the papers presenting the method RITA (Salinesi, Ivankina, and Angole, 2008) were intentionally added to the selected papers.

The primary search (screening) that was based mainly on title, keywords and a succinct read of the abstract might have missed relevant papers related to the topic. Some reuse forms like 'templates' or 'taxonomies' that were discovered through the study were not initially considered in the list of keywords. Moreover, the decision to read or not to read much more than the abstract (for the purpose of selection) strongly depends on the subjective feeling.

There is another threat related to the number of years that are mentioned here: the main searches were based on a defined interval of years. The goal of covering a big interval (2000-2013) turned to be ambitious and difficult to manage. There was a need to restrict the number of papers beyond the selected criteria just to make the process manageable and better reported; this might also induce some bias on the final results.

While executing the research protocol, selecting sources is not an easy and straightforward task. In particular, the choice of quality/selection criteria. For example Quality criterion Q1 (publications containing several unsupported claims or frequently referring to existing work without providing a citation were excluded) may lead to controversial opinions. It depends on subjective judgments by the reviewer, which can only be reduced through feedback from peers. The categorization choices (the map) are another point of discussion. Within the application of the same research protocol, other researchers may decide on a different categorization of the findings.

Results (Conclusion validity)

The results of the SMAP are useful and interesting; however these conclusions are based on sources retrieved in conferences, journals, academic and some industrial reports. It would have been interesting to compare these results with others based on online surveys where real world practitioners are asked about their practices and opinions on security requirements reuse. In addition to that, although there was a careful analysis of the available literature resulting in the presented framework, researches may find that some criteria may have been neglected. Another threat to validity is related to searching exclusively in English writing sources, although it is the largely used language by researchers, but one should pay attention that there are many active communities in other countries who may propose interesting researches related to the topic.

8 CONCLUSION

Over 30 methods to support SRE engineering were presented in this chapter. One can safely say that we are now far away from the first generation of "checklist" based methods as presented by Baskerville (Baskerville, 1993) in the early nineties. A significant number of publications in the requirements engineering community illustrate the steady interest in security requirements engineering during the last two decades. The area of security knowledge reuse is still emerging. One single mapping study can never be able to cover all aspects of existing contributions. Each one can tackle a single aspect.

The contributions of our SMAP can be summarized as follows: (i) a framework was defined for analyzing and comparing the different proposals as well as categorizing future contributions related to knowledge reuse and security requirements engineering; (ii) the different forms of knowledge representation and reuse were identified; and (iii) the situation

of considering domains in SRE process was clarified; (iv) previous surveys were updated. We conclude that most methods should introduce more reusable knowledge to manage security requirements and that the application domains should be considered.

This chapter presented the proceeding and results of a systematic mapping study conducted to get an extensive overview of existing research on knowledge reuse within SRE. The review provides an overview of important existing approaches and tools. More than 30 approaches covering 13 years of SRE practice were analyzed. Our iterative refinement resulted in a final set of five main types of knowledge forms of representation that were (re) used by SRE approaches: (1) security patterns; (2) taxonomies and ontologies; (3) templates and profiles; (4) catalogs and generic models; (5) mixed. For each form of representation, more details were provided about the related SRE approach to it, its (re) use, and the tool support provided. A framework to compare and analyze knowledge reuse and domain consideration in SRE was also defined.

The main goal of our SMAP was to provide a good reference to us in the context of this PhD thesis. This reference can also be useful to other researchers and PhD students to get a clear map on knowledge reuse across SRE and find answers to the different questions on this topic.

This SMAP can also be useful to practitioners (requirements engineers, security officers, security engineers, etc.) who are interested to know what is going in research in terms of SRE. The results of the SMAP can be useful to security architects because they reuse knowledge at corporate level and their responsibilities include leveraging of knowledge reuse. For any given set of requirements, an architect can and should typically identify and evaluate multiple different architectures and architectural mechanisms before selecting what he or she thinks will be the optimum way of fulfilling the requirements. Thus, there are often many ways for an architecture or security team to address a specific kind of security requirement. Therefore, knowing the different methods can make their job easier. These results will also be useful to requirements engineers who have had no training in identifying, analyzing, specifying, and managing security requirements, and requirements teams that do not include subject matter experts in security as it is often the case (Firesmith, 2004).

The SMAP raises new questions that research and industrial communities may face:

At the industrial level, given all these propositions that appeared during the last two decades, what is their maturity (scalability, efficiency) for use in real life industrial environments? The lack of automated support and the fact that many of the SRE methods rely on reusable

knowledge that is not standard remain as issues. The deficiency in automation support may suggest that companies (IT vendors and software editors) still have not invested enough in automation. This leads to the question about how the automation can be made part of existing security technologies that exist already in companies.

On a research level, why do not risk based approaches handle knowledge reuse? Why is there a lack in automated support that handles knowledge reuse for the different SRE methods? One may claim that conceptual methods are often created as part of PhD researches where automation is not always required as part of the dissertation process. However, the research community should be aware of this and should re-focus from method creation to automation and then evaluations to a better assessment of the research contributions.

And, even further in the future, can we imagine a collaborative work between researchers and practitioners for a generalization and unification of all these efforts (like in UML), so that their exploitation in practice and even in academic teaching institutions becomes easier?

The next chapters of this PhD thesis will attempt to reply to part of these questions. The next chapter will concentrate on security ontologies by presenting a survey on the use of ontologies in SRE.

Table 2.3. Domain in SRE

Method	Applied to particular domain?	Domain knowledge
KAOS	Cycab transportation domain	Facts & Assumptions – not domain security requirements
Secure I*	No	No
Secure Tropos	No	No
GBRAM	E-commerce systems	Textual description
Misuse Cases	No	No
Security use cases	No	No
Secure UML	No	No
UMLSec	No	No
Morda	No	No
CORAS	No	Assumptions-not reusable
ISSRM	No	Context-not reusable
CRAC++	No	No
Abuse Frames	No	Causal, lexical, biddable
SREF	No	Trust assumptions - not reusable
Saeki & Kaiya	No	No
SREP	No	Facts & Assumptions - not reusable
MSRA	Ubiquitous technologies	Textual description
SQUARE	No	No
Firesmith	No	No
SIREN	Administration	No
Daramola et al.	No	No
RITA	No	No
Zuccato et al.	Telecommunication services	Textual description
Dritsas et al.	e-health applications	Security Ontology & patterns
Salini et al.	e-voting applications	Security Ontology
Velasco et al.	No	No
Chikh et al.	No	Domain ontology

Table 2.4. Summary of the systematic mapping study (Knowledge reuse)

Form of representation: P=Pattern, Tax=Taxonomy, O=Ontology, C=Catalog, GM=Generic Model, T=Template, Pr= Profile, M=Mixed, - =Nul
Reusable knowledge: T=Threats, C=Countermeasures, A=Asset, O= Organization, G=Goal, V=Vulnerabilities, SR=Security Requirements, - =Nul
Technique: FR=Formal Rules, G=Guidelines, P = Process, Q= Queries, - =Null
Automation: N=No, Y=Yes, - =Null

	Methods (re) using patterns			**Methods (re) using taxonomies or ontologies**								**Methods (re) using templates or profiles**		**Methods (re) using catalogs or generic models**						**Methods (re) using mixed forms of security knowledge**		**Methods not (re) using security knowledge**								
	KAOS	**Secure Tropos**	**Okubo et al.'s method**	**GBRAM**	**Secure Tropos - Si***	**RITA**	**Daramola et al.'s method**	**Dritsas et al.**	**Velasco et al.**	**Salini et al.**	**Chikh et al.**	**Zuccato et al.s method**	**Firesmith**	**Misuse Cases**	**Abuse Frames**	**Security use cases**	**Saeki & Kaiya Method**	**SIREN**	**Secure Tropos**	**SREP**	**SQUARE**	**ISSRM**	**Secure i***	**MSRA**	**Morda**	**CRAC++**	**CORAS**	**UMLSec**	**SREF**	**Secure UML**
Form of Representa-tion	**P**	**P**	**P**	**Tax**	**O**	**O**	**O & T**	**P, O**	**O**	**O**	**O**	**Pr**	**T**	**C& GM**	**GM**	**C&G M**	**C&GM**	**C**	**C& GM**	**M**	**M**	-	-	-	-	-	-	-	-	-
Reusable Knowledge	**T, C**	**C, A**	**A, T, C**	**G V**	**G, C, A**	**T, SR**	**T, SR**	**C, A, V, T**	**A, T, C, V**	**O, T, SR, A, V**	**T, O, V, A, SR**	-	**SR**	**T, SR**	**T, A, V**	**SR**	**T, O, C**	**SR**	**G,T, C**	**T, SR**	**T, SR**	-	-	-	-	-	-	-	-	-
Technique	**FR**	**G**	**P**	-	**FR**	**G**	**Q**	**Q**	**Q**	**Q**	-	**P**	**P**	**G**	**P**	**G**	**FR**	**P**	**G**	-	-	-	-	-	-	-	-	-	-	-
Automation (Reuse tool)	**N**	**N**	**N**	**N**	**N**	**Y**	**Y**	**N**	**N**	**N**	-	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**Y**	**N**	**Y**	-	-	-	-	-	-	-	-	-

Chapter 3

Security Ontologies: A Literature Survey and Classification

With the growing need to implement IT security measures in world-wide corporate environments and the growing application scope, a major obstacle that face ordinary analysts and developers using existing security requirements modeling and analyzing frameworks is the lack of formally and explicitly defined security knowledge and expertise. It becomes also increasingly difficult for them to understand each other due to an imprecisely defined terminology. Problems occur if an Asian employee is drafting a corporate-wide security policy, while his colleague in Russia is misinterpreting the policy, since the terms that were used are not explicitly defined. Some kind of agreed ontology can be used to avoid such inefficiencies (Ekelhart et al., 2007).

In 2003, Marc Donner urged the necessity of having good security ontologies. He argued that too much security terminology is vaguely defined, thus it becomes difficult to communicate between colleagues and, worse, confusing to deal with the people we try to serve: "*What the field needs is an ontology – a set of descriptions of the most important concepts and the relationships among them... A great ontology will help us report incidents more effectively, share data and information across organizations, and discuss issues among ourselves*" (Donner, 2003). The need for security ontologies has been also recognized by the research community in (Denker, 2002).

Since the awareness about security knowledge has grown in the scientific community, many security ontologies have been proposed during the last decade. But there are still many questions around these works: what are the different security ontologies available nowadays? Do they meet the requirements? Do they cover all or some security aspects? Which ontology can I choose as an analyst seeking for security knowledge for the definition of IS requirements? These are the research questions that we attempt to reply to in this chapter.

Because interest in using security ontologies in different fields of research has grown, analysts and researchers may find in this chapter a road map, an overview of what exists in terms of security ontologies. The main objective in this literature survey is to review, analyze, select, and classify security ontologies, as a scope study but with a particular interest in the field of security requirements engineering.

The rest of the chapter is organized as follows: in Section 1, we explain the methodology used in the study, Section 2 includes the survey and classification, and Section 3 recalls related works. Finally, Section 4, the conclusion, raises future perspectives.

1 RESEARCH METHOD

To perform this literature review, we relied on information retrieval and survey methodologies presented in (Levy and Ellis, 2006) (Barnes, 2005) (Rainer and Miller, 2005) (Metcalfe, 2002). Thus, the search strategy undertaken is based on a three-step literature review process. The three steps are: 1) Inputs, 2) Processing, and 3) Outputs. Figure 3.1 provides an overall view of the process undertaken.

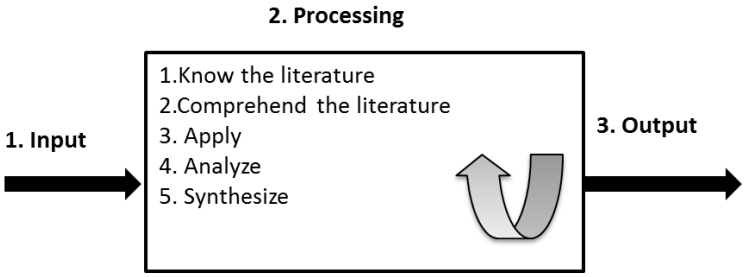


Figure 3.1. The three stages of effective literature review process adapted from (Levy and Ellis 2006)

1.1. INPUT

In any systematic approach, if the system input is either incorrect, of low quality, or irrelevant, the resulted output is going to be ineffective regardless of the quality of the processing stage or, colloquially, garbage-in/garbage-out. In our case, for the input, we gathered, as far as possible, any publication related to the topic. The search was guided by the key words ‘ontologies’, ‘requirements’, ‘security’, ‘concepts’, ‘OWL’, ‘threats’, ‘vulnerabilities’. The search was conducted inside the relevant and known sources of

literature such as Google Scholar, ACM libraries, IEEE digital library etc. About 50 papers were gathered.

1.2. PROCESSING

We performed a first read to get a general idea; 21 papers were discarded at this stage since they were found to be far away from our target objective. A second read was carried out for deeper understanding and analysis of concepts and relations between them. Finally, a quality analysis led us to classify them into different families, and we defined a set of criteria allowing us to compare the researches.

1.3. OUTPUT

The result of the analysis and comparison of the different papers allowed us to define a classification framework of the different security ontologies (see Figure 3.2.). The other output was Table 3.1. The table presents how each retrieved ontology deals with security aspects and requirements.
The next sections present in details the outputs of the literature review.

2 SYNTHESIS AND CLASSIFICATION

It appears that some researchers intend to cover all security aspects and propose general ontologies while others tackle a specific aspect of security; they sometimes refer to previous security taxonomies. In another context, given the increased importance of the World Wide Web in many fields, while security plays a vital role in the success of the Semantic Web, the web community proposed some security related ontologies helping them to define security aspects of web resources and communication. Back to security analysis, some authors proposed related security ontologies by adapting risk analysis: we grouped them in a specific category. Others tried to develop security ontologies for requirements engineering studies, and later with the advancement of security requirements agent models (Secure i* (Liu, Yu, and Mylopoulos, 2003), Secure Tropos (Mouratidis, 2006) (Mouratidis, Giorgini, and Manson, 2003a)), related modeling ontologies were proposed describing the concepts and relationships used. In some cases the security ontologies belong simultaneously to two categories. For example, there are taxonomies for requirements, or web oriented and fairly generic (Vorobiev and Han, 2006). In these cases, we assigned the ontology to the more

dominant field of research. The result is composed of 8 families of security ontologies (Figure. 3.2.), described as follows:

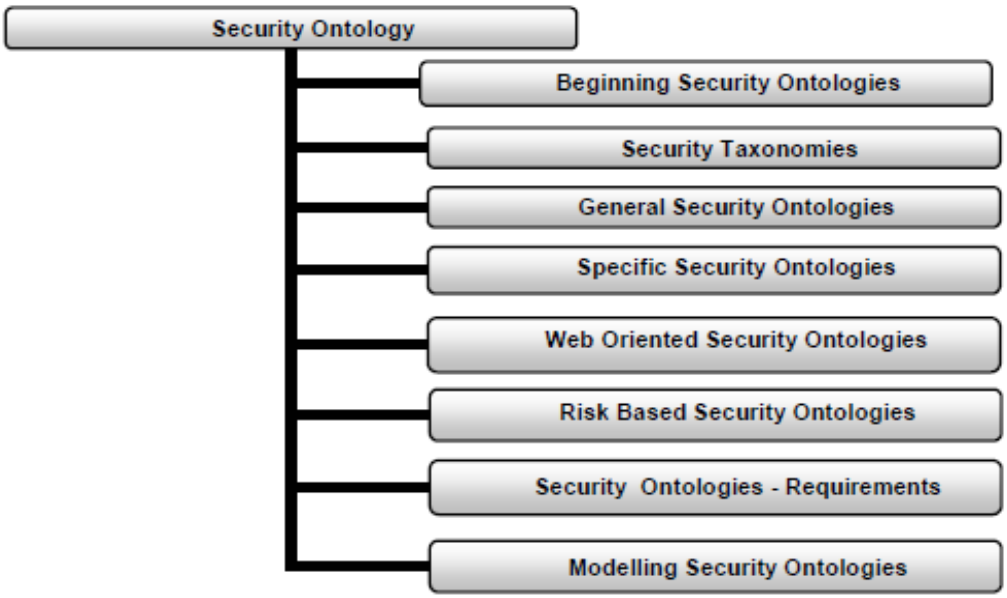


Figure 3.2. Classification of Security Ontologies into 8 families.

2.1. BEGINNING SECURITY ONTOLOGIES

One of the earliest work (back in the nineties) about merging knowledge base and information system management at an early level of development was (Mylopoulos et al., 1990) who proposed a language called Telos for representing knowledge about information systems and illustrated how this language can be applied in developing knowledge bases about software. The knowledge base is divided into four sub-worlds (*subject world, usage world, system world, development world*). Mylopoulos et al. note that Telos users can develop models for the purpose of security specification.

2.2. SECURITY TAXONOMIES

A taxonomy is an ontology in the form of a hierarchy. Whereas ontologies can have any type of relationship between categories, in taxonomies there can only be generalization hierarchies. Taxonomies of security concepts are a common method for sharing security knowledge. There are some interesting taxonomies, which were used later for developing security ontologies:

- (Avizienis et al., 2004) provide a detailed taxonomy that contains classes of *faults*, *fault modes*, classification of *fault tolerance techniques*, and *verification* approaches. In this taxonomy, the main threats to dependability and security are defined as *failures*, *errors*, and *faults*. Avizienis et al. classify the main means to attain security and dependability attributes into *fault prevention*, *fault tolerance*, *fault removal*, and *fault forecasting*.

- (Landwehr et al., 1994) were particularly interested in security flaws. Their taxonomy is based on three basic questions about each observed flaw: genesis (how did it enter the system?), time of introduction (when did it enter the system?), and location (where in the system did it manifest?).

2.3. GENERAL SECURITY ONTOLOGIES

By general ontologies we mean these ontologies which aim at covering all (or most) security aspects:

- Herzog and colleagues (Herzog, Shahmehri, and Duma, 2007) have proposed an OWL-based ontology of information security. They endeavored to deliver an extensible ontology for the information security domain that includes both general concepts and specific vocabulary of the domain, and supports machine reasoning and collaborative development. The proposed ontology is built around the following top-level concepts: *assets*, *threats*, *vulnerabilities* and *countermeasures*. These general concepts together with their relations form the core ontology which presents an overview of the information security domain in a context-independent and application neutral manner. In order to be practically useful, the core ontology is populated with domain-specific and technical vocabulary which constitute the core concepts and implement the core relations. The ontology contains 88 threat classes, 79 asset classes, 133 countermeasure classes, and 34 relations between these classes.

- In the same vein, Fenz and Ekelhart (Fenz and Ekelhart, 2009) have proposed an ontology (500 concepts) that has a similar goal but attempts to cover a broader spectrum: their ontology models a larger part of the information security domain, including non-core concepts such as the infrastructure of organizations. In the high level concepts of the ontology and their relations (cf. Figure 3.3) we find *threat* which gives rise to follow-up threats, represents a potential danger to organization's *assets* and affects specific security attributes (*confidentiality*, *integrity*, *availability*) as soon as it exploits a *vulnerability* in the

form of a physical, technical, or administrative weakness, and it causes damage to certain assets.

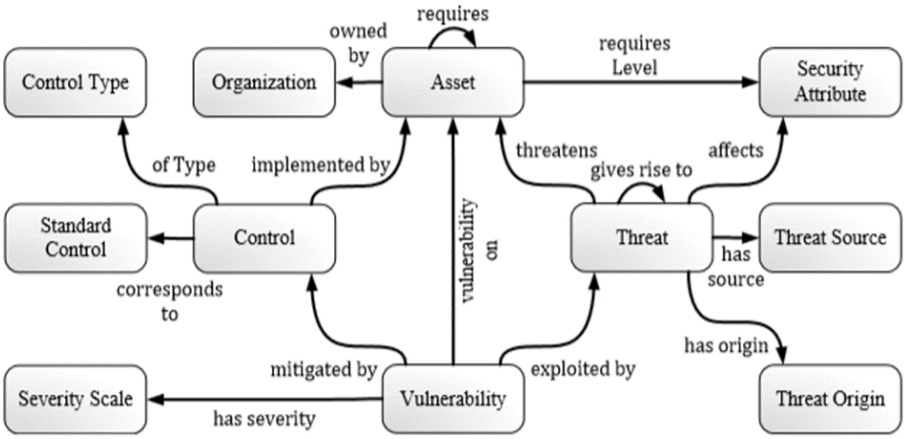


Figure 3.3. Security ontology proposed by (Fenz and Ekelhart 2009)

2.4. SPECIFIC SECURITY ONTOLOGIES

This category gathers the specific domain security ontologies – the ones that describe specified aspects of security such as Session Initial Protocol vulnerabilities, Intrusion detection, etc.

- In (Undercoffer, Joshi, and Pinkston, 2003), the authors propose a data model that characterizes the domain of computer attacks and intrusions as an ontology and implement that data model with an ontology representation language. At the topmost level of the ontology (Figure 3.4.), they define the class Host. The System Component class is comprised of the subclasses (Network, System, Process). The class Attack is described by the properties Directed to, Effected by, and Resulting in. Accordingly, the classes System Component, Input, and Consequence are the corresponding objects. The class Consequence is comprised of several subclasses which include Denial of Service, User Access and Probe. Finally, the class Input is characterized by the predicates Received from and Causing, where Causing defines the relationship between the Means of attack and some input. Received from links Input and Location. The class Location is an instance of System Component and is restricted to instances of the Network and Process classes. Means of attack contains the following subclasses: Input Validation Error, Logic Exploits.

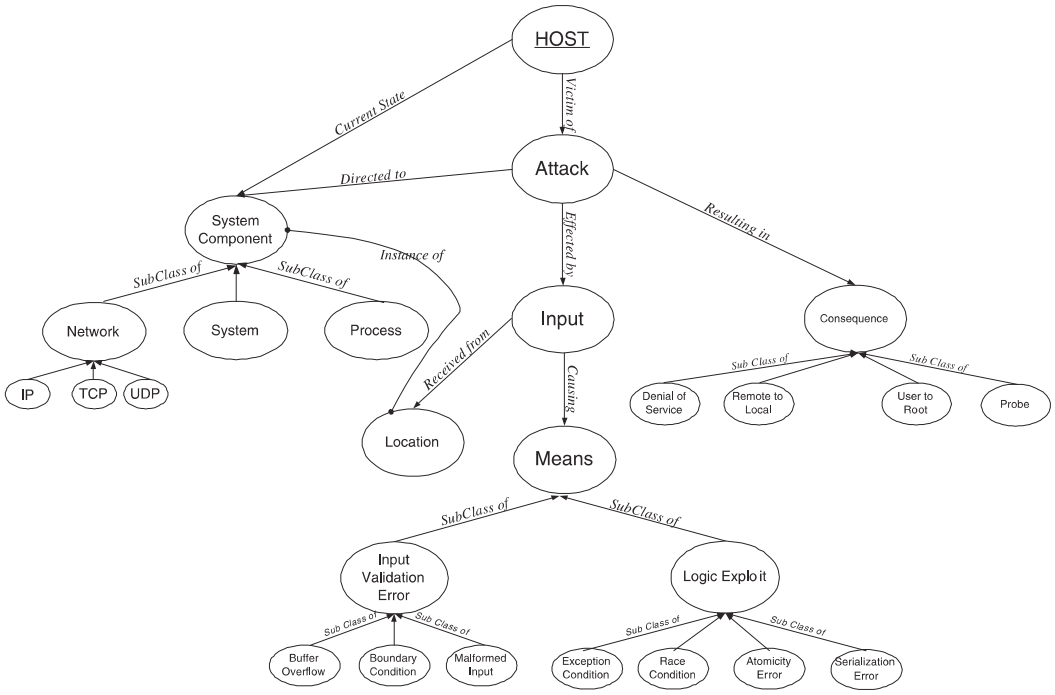


Figure 3.4. Part of the ontology proposed by (Undercoffer, Joshi, Pinkston, 2003)

- (Viljanen, 2005) analyzed thirteen different computational trust models and derived a common vocabulary for describing facts that are considered for trust calculation in the reviewed trust models. The models can be classified as *identity-aware*, *action-aware*, *business value aware*, *capability-aware*, *competence-aware*, *confidence-aware*, *context-aware*, *history-aware* and *third-party-aware* in their input factors. The trust ontology comprises many ontological structures; *trust* is a relationship between two principals, the subject, *trustor*, and the target, *trustee*.

- Voice over IP (VoIP) telephony services suffer from various types of attacks and vulnerabilities, mainly due to the utilization of an open environment, the Internet. (Geneiatakis and Lambrinoudakis, 2007) propose an ontology for SIP-VoIP based services. This ontology can be applied either to find a countermeasure against attacks on SIP based VoIP services or for testing the security robustness of SIP-VoIP (Session Initial Protocol-VoIP) infrastructure. The ontology contains two main concepts *SIP_attack* and *SIP_message*. Specifically any SIP attack employs a SIP message that is forwarded to a target node trying to cause a specific consequence. The SIP_attack is directed by a target and causes a consequence. It has two subclasses: *malformed* and *flood*.

2.5. WEB ORIENTED SECURITY ONTOLOGIES

Some works addressed both the security community and the semantic web community.

- Denker et al in (Denker et al., 2003) (Denker, Nguyen, and Ton, 2004) (Denker, Kagal, and Finin 2005) develop several ontologies for security annotations of agents and web services, using DAML (DARPA Agent Markup Language) and later OWL (Web Ontology Language). The defined ontology is composed of two sub-ontologies: "*security mechanisms*" which capture high-level security notations and "*credential*" which defines authentication methods. The goal of these ontologies is to enable high-level markup of Web resources, services, and agents while providing a layer of abstraction on top of various web service security standards. These ontologies represent well-known security concepts and enable their users to interconnect security standards.

- The NRL Security Ontology proposed in (Kim, Luo, and Kang, 2005) is organised around seven separate ontologies (*Main Security Ontology*, *Credential Ontology*, *Security Algorithms Ontology*, *Security Assurance Ontology*, *Service Security Ontology*, *Agent Security Ontology*, *Information Object Ontology*). Three of them are based on existing based ontologies in DAML: firstly, "*Service security ontology*", which describes security annotation of semantic web services; secondly, "*Agent security ontology*", which enables querying of security information; and finally "*Information object ontology*", which describes security of input and output parameters of web services. The four remaining ontologies are as follows: "*Main security ontology*", describes security protocols, mechanisms and policies; "*Credentials ontology*", specifies authentication credentials; "*Security algorithms ontology*", describes various security algorithms; and "*Security assurance ontology*", specifies different assurance standards.

- Artem Vorobiev and Jun Han proposed a security attack ontology for Web service (Vorobiev and Han, 2006). The ontology brings together a set of attacks (*attacks on Web services*, *probing attacks*, *CDATA Field attacks*, *WS DoS attacks*, *WS DoS attacks*, *Application attacks*, *SOAP attacks*, *XML attacks*, *semantic WS attacks*).

2.6. RISK BASED SECURITY ONTOLOGIES

Recent trends in security methodologies tend to consider that the best approach of security consists in starting from a risk analysis. It allows the experts to adapt the security solutions to the actual risk, leading to a more effective security plan.

- (Ekelhart et al., 2007) proposed a security ontology framework based on four parts: the first part is the security and dependability taxonomy from (Avizienis et al., 2004), the second part presents the underlying risk analysis methodology, the third part describes concepts of the IT infrastructure domain and the fourth part provides a simulation enabling enterprises to analyze various policy scenarios. The ontology 'knows' which threats endanger which assets and which countermeasures could lower the probability of occurrence, the potential loss or the speed of propagation for cascading failures.

- (Assali, Lenne, and Debray, 2008) proposed to develop a knowledge base containing ontologies for the analysis of industrial risks describing concepts used for the achievement of a risk analysis.

2.7. (SECURITY) ONTOLOGIES FOR SECURITY REQUIREMENTS

Some papers refer to ontologies in order to cope with the definition of security requirements:

- (Dobson and Sawyer, 2006) propose an ontology of dependability by merging two conceptualisation models (IFIP model: proposed by the IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance & UMD model: Unified Model of Dependability). Some of the IFIP attributes are themselves goals of security (Availability, Integrity, Maintainability, and Confidentiality). The ontology covers some security aspects such as *Failure*, Dependability Threat (*Error*, *Fault*), Dependability Attributes (*Availability*, *Integrity*, *Confidentiality*…)

- (Tsoumas and Gritzalis, 2006) define a security ontology using OWL and propose the security framework of an arbitrary information system which provides security acquisition and knowledge management. Tsoumas et al. have used *Asset*, *Stakeholder*, *Vulnerability*, *Countermeasure* and *Threat* concepts in the construction of the security ontology. The security ontology acts as a container for the IS security requirements ("What" part).

- In (Karyda et al., 2006), the authors use OWL to propose a security ontology with which to develop secure applications. The proposed ontology is formed of "*assets*" (data asset, hardware data,...), "*countermeasures*" (identification and authentication, network management, auditing services, physical protection, etc.), "*objectives*", "*persons*" (insider stakeholder, attacker,...) and "*threats*" (errors, attacks, technical failures, etc.). They validate the defined ontology using nRQL queries in order to demonstrate that their ontology can be used in various contexts. They apply it to e-government scenarios: e-tax and e-voting.

- (Firesmith, 2005) presents a taxonomy of safety-related requirements: "*Safety requirements*" are requirements obtained from threats analysis. "*Safety-significant requirements*" include non-safety requirements that can cause hazards and safety incidents. "*Safety constraints*" are constraints that directly impact safety and are derived from laws, policies, standards, and industrial practices. "*Safety system requirements*" specify aspects of the primary system.

2.8. SECURITY MODELLING ONTOLOGIES

Even if authors present them as ontologies, they mainly describe meta-models. While the previous ontologies include security specific concepts such as threat, attack, vulnerability, these ontologies include security related concepts for modelling requirements and the dependencies between them such as relationship, proposition, and situation.

- In (Mouratidis, Giorgini, and Manson, 2003b), first, the concept of *security constraint* is introduced, as a separate concept, next to the existing concepts of Tropos. Secondly, existing concepts such as goals, tasks, resources, are defined with and without security in mind. For example a goal should be differentiated from a secure goal, the latter representing a goal that affects the security of the system. Thirdly, security-engineering concepts such as *security features*, *protection objectives*, *security mechanisms* and *threats*, which are widely used in security engineering, are introduced in the Tropos ontology, in order to make the methodology applicable by software engineers as well as security engineers.

- (Massacci et al., 2011) propose an extended ontology for security requirements. The very top of the taxonomy is adapted from DOLCE, a foundational ontology intended to account for basic concepts that underlie natural language and human cognition. Lower levels of the taxonomy include concepts from i*, problem frames and argumentation frameworks, with security concepts occupying the lowest strata of the taxonomy. Let's list some of their proposed concepts: Objects (*Proposition*, *Situation*, *Entity*, *Relationship*) – Entities (*Actor*, *Action*, *Process*, *Resources*, *Assets*) – Relationships (*do-dependency*, *can-dependency*, *trust-dependency*) from SI* – Propositions (*Fact*, *Claim*, *Argument*, *Domain-Assumption*, *Quality Proposition*, *Goal*).

Thus many papers propose security ontologies composed of different but related concepts aiming at common or different objectives. The following section compares and evaluates them.

3 DISCUSSION AND EVALUATION

In this section, we compare security ontologies and try to evaluate to which extent they cover security requirements and thus can be used in requirement engineering.

(Mylopoulos et al., 1990) did not literally propose a security ontology nor an ontology, but a basic taxonomy composed of four sub-worlds. The authors note that users of Telos (the proposed language for developing the knowledge base and the sub-worlds) have developed models for the purpose of security specification but did not detail the underlying models.

In the family of security taxonomies, (Avizienis et al., 2004) proposed a detailed taxonomy of security and dependability. But this taxonomy fails to cover techniques for protecting confidentiality, establishing authenticity, analysing issues of trust and the allied topic of risk management. Some important security elements are not addressed, such as vulnerabilities and assets. The taxonomy doesn't deal with any use for requirements.

The two main limit in the taxonomy of (Landwehr et al., 1994) is that it is too basic, focused on some flaws in operating systems only, far from many kinds of security flaws that might occur in application programs for database management, word processing, electronic mail, and so on. The study needs to be updated with recent work. Flaws in networks and applications are becoming increasingly important, and the distribution of flaws among the categories they have defined may not be stationary. The taxonomy of (Landwehr et al., 1994) focused on a special kind of threats and does not address any countermeasure or related vulnerability.

The two general security ontologies (Herzog, Shahmehri, and Duma, 2007) and (Fenz and Ekelhart, 2009) are both interesting contributions but neither of them is complete. While the first one seems simple and clearer, the second is much richer but more complex. Fenz et al. better cover asset concepts, while Herzog et al. focus on threat concepts. Fenz's main contribution consists of the organisation concepts, clearly absent from Herzog. Herzog's countermeasures tend to be technical whereas Fenz's are both business and technical. The advantage of being generic and capturing most security aspects leads also to drawbacks since the ontologies lack in specificity that the specific security domain ontologies (Undercoffer, Joshi, and Pinkston, 2003), (Viljanen, 2005), (Geneiatakis and Lambrinoudakis, 2007) provide, and vice-versa. Neither (Herzog, Shahmehri, and Duma, 2007) nor (Fenz and Ekelhart, 2009) ontologies were used for requirements definition and analysis, but both, combined with the specific ontologies, can be a very good source of security knowledge for requirements. The general ontologies offer generic concepts of security objectives, assets,

vulnerabilities, countermeasures, threats, etc. while the rest offers more specific threats concepts (computer attacks and intrusions in (Undercoffer, Joshi, and Pinkston, 2003), for example).

The security ontologies developed in the semantic web area are not negligible. The ontology of (Kim, Luo, and Kang, 2005)looks like a generic security ontology from a first sight with its seven sub-ontologies. However it does not cover some aspects like vulnerabilities and assets or organisation, or even threats. Nevertheless, in a web sharing community, where both resource requestors and providers have security requirements, (Kim, Luo, and Kang, 2005) proposed a matching algorithm that facilitates mapping of higher level (mission-level) security requirements to lower-level (resource level) capabilities using the ontology. In a very similar previous work by Denker et al. (Denker et al., 2003) (Denker, Nguyen, and Ton, 2004) (Denker, Kagal, and Finin, 2005), the proposed ontology fails to consider vulnerabilities, assets and threats; but a reasoning engine matches between the request requirements and the capabilities of the potential web service whose requirements need to be satisfied by the capabilities specified in the request.

The risk based security ontologies we found in (Ekelhart et al., 2007) and (Assali, Lenne, and Debray, 2008) could be useful for a risk based requirement analysis like (Herrmann et al., 2011) or (Mayer, Rifaut, and Dubois, 2005). However, to the best of our knowledge, there were no propositions combining both sides.

In the context of requirements engineering some ontologies were proposed, but unfortunately none of them is associated to a methodology describing how to use them for requirement definition. Dobson and Sawyer's ontology (Dobson and Sawyer, 2006) concentrates on few threat concepts, and neglects many other aspects of security. (Tsoumas and Gritzalis, 2006), in addition to their ontology, provided a framework, but don't indicate any detailed mechanism on how to use the ontology for requirement collection. The main lack of (Karyda et al., 2006) is the absence of vulnerability related concepts, although they propose many examples of queries on the ontology, that provide answers to the developer in an e-government application.

Finally, the security modelling ontologies, which are more security modelling oriented (relationship, entity...) than security concept oriented (assets, threats ...) might be useful for constructing security requirements models like Secure i* and Tropos. A limitation common to all the ontologies we have been facing is that they are described in papers but are not available on the Internet, which makes their use difficult.

We summarise this analysis and evaluation in the table 3.1. The rows are the families of security ontologies. The columns list the aspects related to security (objectives, assets, vulnerabilities, threats, countermeasures and organisation). The last column in the table evaluates the link between the ontology and requirements definition. A black dot measures to which extent the security ontology covers this specific aspect of security, and how this particular security ontology deals with requirements. We used a dash for absence of use, as follows: How does the ontology cover this concept of security? How does this security ontology proposal deal with requirements (last column)?

-: absent ●: very few ●●: few ●●●: much ●●●●: very much

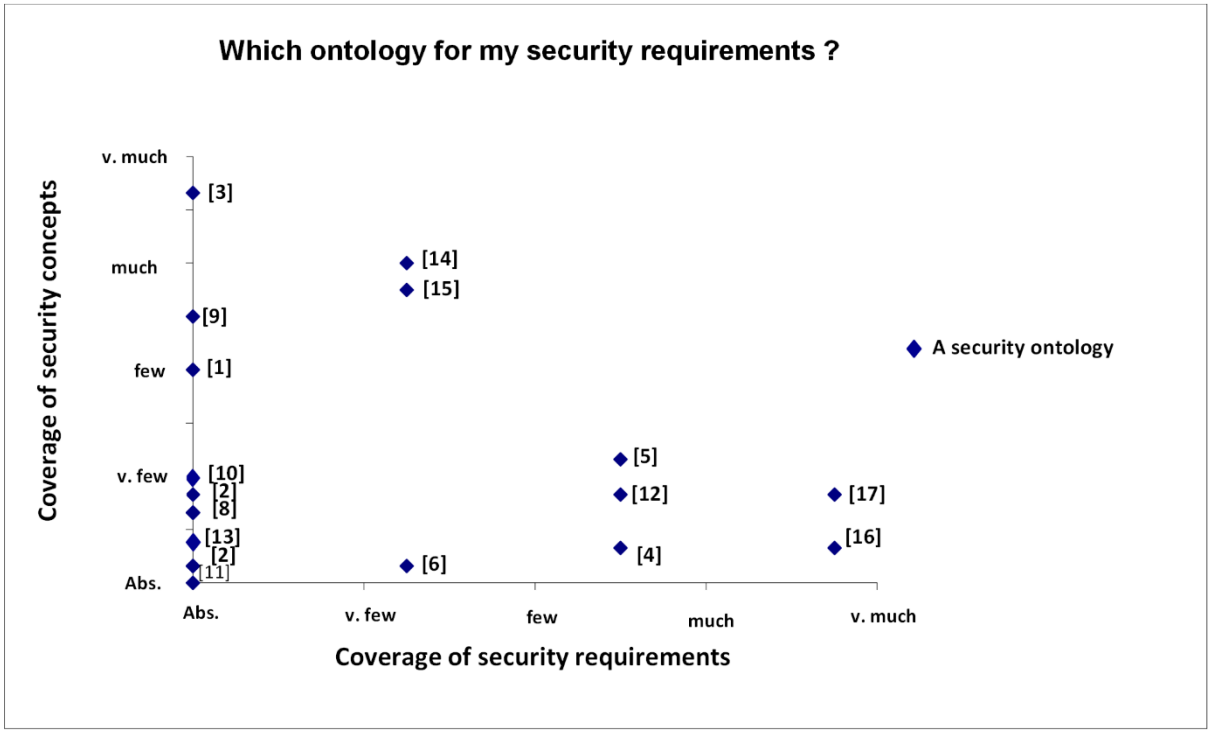
To complete the study we drew up a graph that represents roughly, for each security ontology, how much it deals with requirements (x-axis) and how much it covers security concepts (y-axis). The graph in Figure 3.5 clearly reveals a gap between the two fields. There is not a perfect ontology that covers lots of security aspects and, at the same time that can be used in the definition for security requirements

Table 3.1. Summary of security ontologies of the study

Family	Security Ontology[1]	Security Objective[1]	Assets[1]	Vulnerabilities[1]	Threats[1]	Countermeasures[1]	Organization[1]	**Requirements**[2]
Beginning	(Mylopoulos et al., 1990)	-	-	-		-	•	•
Security taxonomies	(Avizienis et al. 2004)	•••	-	-	••	••••	-	-
	(Landwehr et al. 1994)	-	-	-	••••	-	-	-
General	(Herzog, Shahmehri, et Duma 2007)	••	•••	••	•••	•••	••	-
	(Fenz and Ekelhart 2009)	••••	••	••••	••••	••••	••••	-
Specific	(Undercoffer, Joshi, Pinkston 2003)	-	•	•	••	-	-	-
	(Viljanen 2005)	•	-	-	-	-	-	-
	(Geneiatakis & Lambrinoudakis 2007)	••	-	-	•••	-	-	-
Risk based	(Ekelhart et al. 2007)	••	-	-	••••	•••	•••	-
Web oriented	(Denker et al. 2003) (Denker, Nguyen, Ton 2004) (Denker, Kagal, Finin 2005)	•••	-	-	-	••	-	••
	(Kim, Luo, Kang 2005)	••••	-	-	-	•••	-	••
	(Vorobiev & Han 2006)	-	-	-	••••	-	-	-
For security requirement	(Dobson & Sawyer 2006)	••	-	-	••	-	-	-
	(Tsoumas & Gritzalis 2006)	-	•••	•••	••	•••	•	•
	(Karyda et al. 2006)	••	•••	-	•••	•	••	•
	(Firesmith, 2005)	-	•	-	•	-	-	••
Modelling	(Mouratidis, Giorgini, Manson 2003b)	•	-	-	-	-	•	•••
	(Massacci et al. 2011)	••	••	-	•	-	-	•••

[1] How does the ontology cover this concept of security? [2] How does this security ontology deal with requirements? (Last column)

- : Absent, •: Very few, ••: Few, •••: Much, ••••: Very much



Legend of the table:

[1]	(Ekelhart et al. 2007)
[2]	(Dobson & Sawyer 2006)
[3]	(Fenz and Ekelhart, 2009)
[4]	(Firesmith, 2005)
[5]	(Kim, Luo, and Kang 2005)
[6]	(Mylopoulos et al. 1990)
[7]	(Avizienis et al. 2004)
[8]	(Landwehr et al. 1994)
[9]	(Herzog, Shahmehri, and Duma 2007)

[10]	(Geneiatakis & Lambrinoudakis 2007)
[11]	(Assali, Lenne, Debray 2008)
[12]	(Denker, Nguyen, Ton 2004)
[13]	(Vorobiev & Han 2006)
[14]	(Tsoumas & Gritzalis 2006)
[15]	(Karyda et al. 2006)
[16]	(Mouratidis, Giorgini, Manson 2003b)
[17]	(Massacci et al. 2011)

Figure 3.5. Qualitative evaluation of security ontologies coverage of security concepts.

4 SURVEYS ON SECURITY ONTOLOGIES

While many security ontologies have been proposed, few surveys have been attempted. The only ones we can cite here are (Blanco et al., 2008), (Elahi et al., 2011) and recently (Nguyen, 2011). (Elahi et al., 2011) was not primarily about ontologies, but they mentioned some security ontologies and taxonomies in their state of art of security requirements. (Nguyen, 2011) proposed a survey of general ontologies for information systems encompassing some few security ontologies. Blanco et al. (Blanco et al, 2011) is an interesting review and comparison of security ontologies that helped us in our study. However, since 2009 other ontologies have been proposed, indicating a need for updating. Moreover, Blanco et al. organized the existing ontologies under four categories (general security ontologies, applied to a specific domain, theoretical works, semantic web-oriented). Our aim was to extend this classification to additional categories and to update their surveys with recent literature contributions.

5 CONCLUSION

Let us come back to our main question: which security ontology for my requirements? This study has shown the existence of considerable work around security ontologies; several ontologies have been proposed. We classified them into eight families (theoretical basis, security taxonomies, general, specific, risk based, web oriented, requirements related, modelling). This classification extends the previous works, which were limited to two, three, or four families at best.

Our analysis has also shown that the existing security ontologies vary a lot in the way they cover security aspects; we tried to analyse how each ontology covers each aspect of security (objectives, assets, vulnerabilities, threats, countermeasures, and organisation). Moreover, we studied whether the proposed security ontology can be used for requirements definition and the degree of this use.

The study revealed a gap between the fields of security requirement engineering and ontologies, and thus a new area of research to explore.

We believe that this work can be improved; the classification needs to be extended. We need sub-categories for each family of security ontologies. The study revealed that there are still

important issues to be addressed in the adaptation of ontology-based requirements engineering techniques to security requirements engineering.

This study allows us to assert that the challenges facing software security is the lack of an easily accessible large common body of security knowledge. Although much security ontologies are available, they all fall short in completeness and suitable granularity. It also remains difficult for designers to extract relevant pieces of knowledge to apply to their specific design or requirements related decision making situations.

The objective in the next chapters of this PhD thesis is to unify the knowledge provided in the proposed security ontologies and explore the techniques and mechanisms for the best use of this knowledge for security requirement definition.

Part 2

Main contribution

Chapter 4

AMAN-DA
A method for domain specific security requirements engineering

This chapter provides the overview of the ontology-based security requirements elicitation method proposed in this PhD thesis: AMAN-DA[3]. The chapter presents the main elements that construct the method, the inputs and the expected outputs. AMAN-DA takes as an input security, domain ontologies and stakeholder's security goals. It produces as an output security requirements models and textual security requirements specifications. This is done thanks to a set of ontology-based rules and mechanisms. In this chapter, a running example related to web applications domain is presented.

1 PROPOSITION

AMAN-DA proposes to use in combination two kinds of ontologies: a security ontology that embeds security specific knowledge, and a multi-level domain ontology that encompasses domain specific knowledge. The expected outcome is that the security requirements resulting from the combined use of both ontologies will be more domain-specific. Figure 4.1. gives an overview of the proposed method.

AMAN-DA is generic in the sense that different domain ontologies can be used with it. However it is domain specific when it is applied in the sense that during its application only one domain ontology is used. The method relies on a collection of heuristic rules that extract relevant security and domain knowledge from ontologies (red arrows in Figure 4.1.). In addition to the two ontologies (security and domain), the inputs are security goals (such as the ones that are captured during interviews with stakeholders). The output is a specification of

[3] *AMAN* (أمان) is the Arabic name for security. DA is for Domain of Application. The name was chosen to refer to security requirements engineering for domains of application. Ironically, the name AMANDA in Latin means "worthy of love"!

security requirements well-formalized with Secure Tropos model (SecTro) (Mouratidis and Giorgini, 2007) in addition to a specification of textual security requirements.

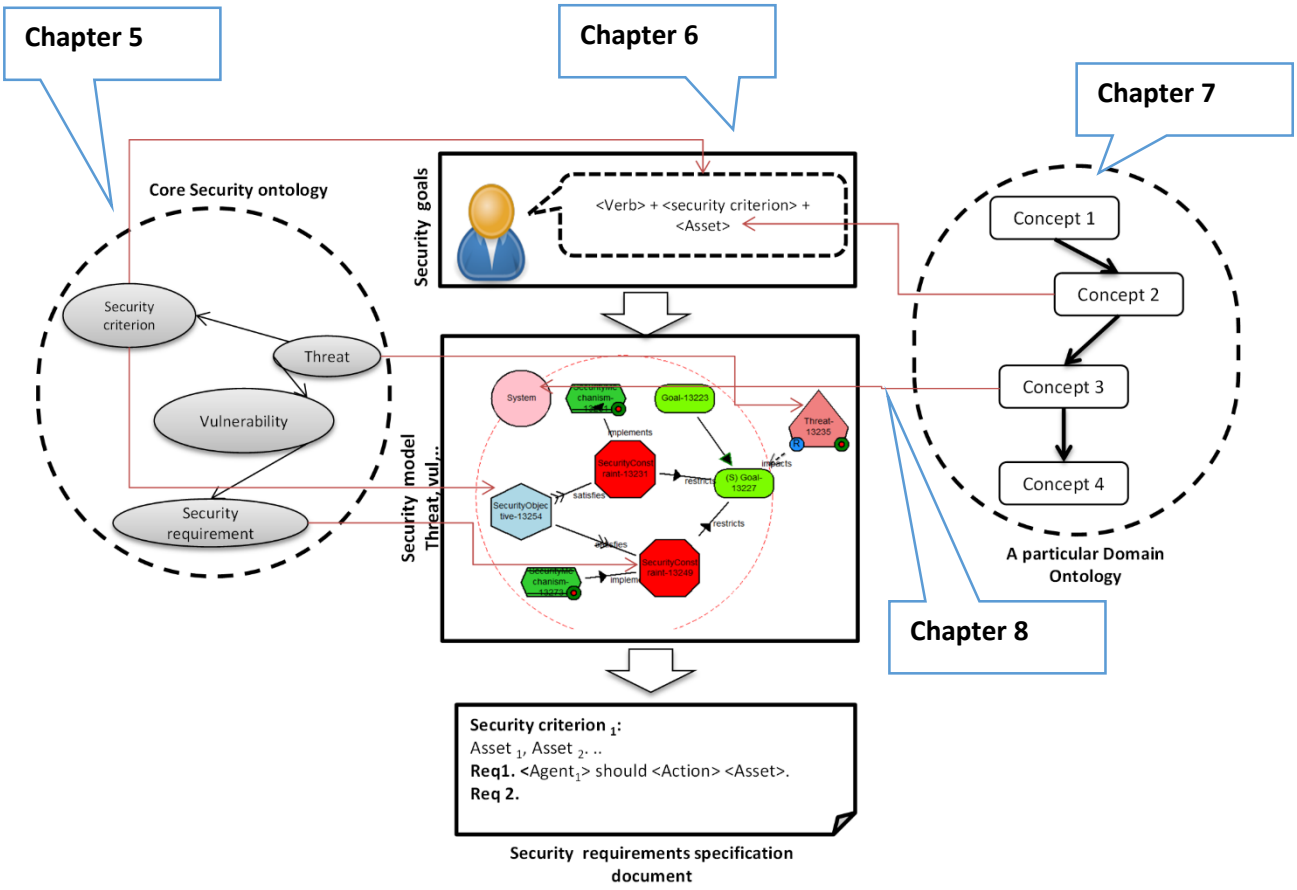


Figure 4.1. Overview of AMAN-DA

The choice of Secure Tropos was motivated by the fact that it is one of the richer modeling frameworks in terms of security concepts (according to our literature review presented in Chapter 2).

The originality of the method lies: (a) in the fact that the combination of the security ontology and domain ontologies is not achieved a priori, but at runtime, while the method is applied; (b) in the genericity of the method, in the sense that it is designed to be used with a generic security ontology and any domain ontologies, as long as they embed some expected knowledge;(c) the defined rules and presented algorithm allows the method to automatically exhibit an appropriate ontological semantics (security and domain specific) to the

requirements engineer (agents, objects, threats, security requirements…);(d) the automatic generation of Secure Tropos models and textual security requirements.

The null hypothesis H0 is that "Using domain ontologies in addition to a security ontology to guide security requirements elicitation does not make a difference with respect to discovering requirements with a security ontology only". What we want to demonstrate is that such a method will be more efficient, and easy to use by requirements engineers (and security analysts) to produce security requirements (models and texts) that are more domain specific.

The next chapters will present each element of the method in more details:

- Chapter 5 presents the core security ontology.
- Chapter 6 presents a specification of security goals and security requirements.
- Chapter 7 presents the multi-level domain ontology.
- Chapter 8 presents the mechanisms and rules that put everything together to produce Secure Tropos models and textual security requirements specifications.

2 RUNNING EXAMPLE

Let us consider a requirements engineer (Zach), who prepares specifications for different products. In these specifications, he usually focuses on functional requirements. The limitations of his tacit knowledge about security and of his primitive knowledge about the domain for which he elicits security requirements make his resulting security requirements poor and too generic. In addition, a short interview with stakeholders is not enough to know all the domain agents and objects that are present in the project. Sometimes, such engineers copy/paste the security requirements from other projects in an ad-hoc way, which is relatively time consuming and disorganized way of working.

This time, Zach is asked to elicit security requirements of a web application of an online shopping company. The engineer wants to i) capture the stakeholders' goals in a quick manner, ii) produce well-formed Secure Tropos model, iii) produce the corresponding well–formed textual security requirements, and iv) throughout the elicitation process, he wants to get access to domain knowledge related to online shopping without having to refer to the stakeholder.

The next chapters will present each element of the method, and the first one is the core security ontology.

Chapter 5

A core security ontology for security requirements engineering

The research community of information system security (Donner, 2003) urged the necessity of having a good security ontology to harmonize the vaguely defined terminology, leading to communication troubles between stakeholders. The benefits of such security ontology are manifold: it will help requirements engineers report incidents more effectively, reuse security requirements of a same domain and discuss issues together (Souag, 2012).

AMAN-DA proposes a core security ontology that considers the descriptions of the most important concepts related to security requirements and the relationships among them. "*Core*" refers to the union of knowledge (high-level concepts, relationships, attributes) present in other security ontologies proposed in the literature.

This proposition is not the first one intending to tackle these issues; however, its particular contribution is to offer an answer to the requirements engineering community. Indeed, some security requirements engineering methods use ontologies (as reported in chapter 2), such as RITA (Salinesi, Ivankina, and Angole, 2008), Daramola et al. (Daramola, et al., 2012a), Velasco et al. (Velasco et al., 2009) and Salini et al. (Salini and Kanmani, 2012a). The problem is that most of these studies are limited to a particular field of use, and the perimeter they cover remains relatively small compared to the scope of "security" (Souag, Salinesi, and Comyn-Wattiau, 2012) (Blanco et al., 2008). As (Massacci, Mylopoulos, and Zannone, 2007) claim "*Although there have been several proposals for modeling security features, what is still missing are models that focus on high-level security concerns without forcing designers to immediately get down to security mechanisms*". Meta-models can be useful since they provide an abstract syntax of security concepts. However, we believe that ontologies can be a better option since they allow representing, accessing, using and infering about that knowledge in order to develop methods, techniques, and tools for security requirements analysis.

According to (Blanco et al., 2011), a good security ontology should *inter alia*, include static knowledge (concepts, relationships and attributes), and dynamic knowledge (axioms). It must

be reusable (commented in natural language, and formalized in a standard language).

The main objective of this chapter is to address the following research question: What are the concepts and relations that need to be present in a core security ontology?

This ontology should make it possible to:

- Consolidate and capitalize the knowledge of the research community, by creating an entry point for the various existing ontologies in the literature.
- Create a generic platform of different security concepts (threats, risks, requirements, etc.). This ontology will harmonize the semantics of existing security ontologies.
- Create a source of reusable knowledge for the elicitation of security requirements in various projects.

The rest of the chapter is organized as follows: Section 1 presents the construction of the ontology, its concepts and relationships. Section 2 reports the evaluation of the proposed ontology. Finally, Section 3 concludes the chapter and describes future work directions.

1 A CORE SECURITY ONTOLOGY FOR SECURITY REQUIREMENTS ENGINEERING

This section presents the main contribution of the chapter, a core security ontology to be used particularly for the security requirements elicitation process. The method for constructing the security ontology is adapted from ontology construction methods proposed by Fernandez et al. (Fernández-López, Gómez-Pérez, and Juristo, 1997), mixed with key principles of the ones proposed by Jones et al. (Jones, Bench-Capon, and Visser, 1998).

The construction process contains six main steps: objective, scope, knowledge acquisition, conceptualization, implementation, and validation. The objective behind the ontology construction must be defined in the beginning, including its intended uses, scenarios of use, end-users, etc. The scope stipulates the field covered by the ontology. The knowledge acquisition step aims at gathering from different sources the knowledge needed for the ontology construction. In the step of conceptualization, the knowledge is structured in a conceptual model that contains concepts and relationships between them. Ontology implementation requires the use of a software environment such as Protégé[4]; this includes

[4] http://protege.stanford.edu/

codifying the ontology in a formal language (RDF[5] or OWL[6]/XML[7]). Finally, the validation step guarantees that the resulting ontology corresponds to what it is supposed to represent. The details about how the first five steps were applied to construct our ontology are presented in the following sub-sections and the last step is detailed in Section 2.

1.1. OBJECTIVE

The main objective of the target ontology is to provide a generic platform containing knowledge about the core concepts related to security (threats, vulnerabilities, countermeasures, requirements, etc.). This ontology will be a support for the elicitation of security requirements and the development of SRE methods and tools; it will be in particular used in the context of our proposed method (AMAN-DA). The ontology will be a meta-view for the different security ontologies in the literature. It should harmonize the security terminology spread in these ontologies and help requirements engineers communicating together.

1.2. SCOPE OF THE ONTOLOGY

The ontology covers the security domain in its high level aspects (threats and treatments) as well as its organizational ones (security procedures, security management process, assets, and persons). The reader will find details on all security concepts covered by the ontology in section 1.4 below on.

1.3. KNOWLEDGE ACQUISITION

The acquisition of the security knowledge started from standards (e.g. ISO27000[8]). Other knowledge acquisition sources were the different security ontologies and security models that exist in the literature. We analyzed about 20 security ontologies, based on our previous literature survey (Souag, Salinesi, and Comyn-Wattiau, 2012). These ontologies are of various levels (general, specific, for a particular domain). Relevant concepts and relationships were extracted through a systematic analysis of the security ontologies. Table 1 in the appendix B presents part of them (13 ontologies). A brief descriptions of some of those ontologies are proposed in the following:

[5] http://www.w3.org/RDF/
[6] http://www.w3.org/TR/owl-features/
[7] http://www.w3.org/XML/
[8] http://www.iso.org/iso/home/standards/management-standards/iso27001.htm

- Fenz et al. (Fenz and Ekelhart, 2009) have proposed an ontology to model the information security domain. We reused some concepts and relationships of that ontology, in particular the ones related to the infrastructure of organizations (assets, organization), the relationships between threats and assets, and between threats and vulnerabilities. Fens et al.'s ontology was a rich source of threats, vulnerabilities and standard controls that have been reused to define our security requirements.
- The ISSRM model (Mayer, 2012) (top left in Figure 5.1.) was defined after a survey of the risk management standards, security related standards, and security management methods. The three groups of concepts proposed in the ISSRM model (asset related concepts, risk related concepts, and risk treatment related concepts) were used to define the three dimensions of the ontology (organization, risk, treatment).
- Velasco et al.'s security requirements ontology (Velasco et al., 2009) was useful to define the security requirements in our ontology.

Figure 5.1 schematizes the knowledge acquisition step and part of the conceptualization phase, starting with the knowledge sources (the different ontologies), the concept alignment, and the conceptualization with the help of experts and documents. The concepts of the resulting ontology were derived from the alignments of the different security ontologies in the knowledge acquisition step.

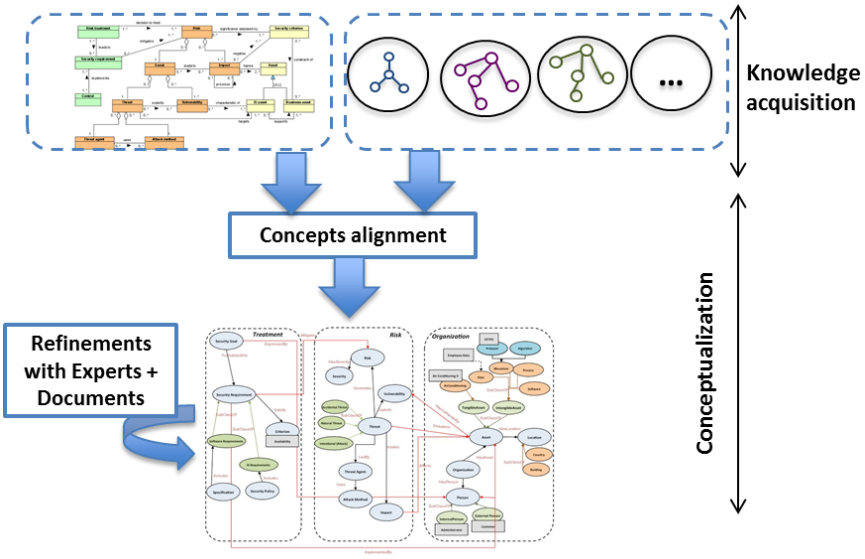


Figure 5.1. Knowledge acquisition and conceptualization phases.

1.4. CONCEPTUALIZATION

Based on the outcomes of the knowledge acquisition step, concepts were organized and structured in a glossary. Various relationships among these concepts were considered, and then were put together in a conceptual model of the ontology (Figure 5.2.), easy to understand, independently of any implementation language. The names of the concepts and the relationships of the security ontology proposed in this paper were chosen according to the number of occurrences of names in the source ontologies (Table 1 in the appendix B). If a concept has different names in the ontologies (e.g. impact or consequence, attack method or deliberate attack, or SessionIP attack); the most generic or easiest to understand name was chosen (here, impact, attack method). Some security experts were consulted through email exchanges, phone and direct discussions to validate our choices. The experts acknowledged most of the concepts and relationships between them. Some refinements in the ontology were performed after discussion with them. For example, the concept of "Attack" was removed; the experts consider it as an Intentional Threat. Discussions also clarified the difference between the concepts of "Security Goal", "Security Criterion", "Security requirement" and "Control". These concepts are frequently mixed up in the security requirements elicitation phase and the difference between them is often not easy to capture. The concepts were organized around three main dimensions. The latter are: *Risk dimension*, *Treatment dimension*, and *Organization dimension*. The *Risk dimension* represents the "dark" face of security; it gathers concepts related to 'threats', 'vulnerabilities', 'attacks', and 'attackers'. *Treatment dimension* is concerned with concepts related to the necessary treatments to overcome risks. The concepts are 'security goals', 'requirements', 'controls', and 'security policies'. Finally, security is a multifaceted problem; it is not only about technical solutions or single assets, but also about the environment where threats appear and arise. That is why the *Organization dimension* is considered. This dimension relates to concepts such as 'person', 'location', 'assets', and 'organization' that must be analyzed and on which assumptions must be made in a security requirements elicitation process. Some ontologies covered only the dimension treatment (Velasco et al., 2009). The security ontology proposed by (Fenz and Ekelhart, 2009) groups concepts into three sets (security, enterprise and location). The classification into these three dimensions that was inspired from the model proposed by Mayer et al. (Mayer, 2012) (organization, risk and treatment) helps in organizing the knowledge related to security. The concepts and relationships of the ontology are described in the following sub-section.

A. Concepts of the security ontology

The following summarizes the different concepts identified for the ontology with their respective descriptions. These general concepts together with their relations constitute the ontology, which presents an overview of the information security in a context-independent manner. In the following, we describe the concepts dimension by dimension.

a) Organization dimension

This dimension includes the concepts related to the organization, its assets and its environment. The concepts are:

- Organization: a structure including human, hardware, and software resources (assets).
- Person: Represents human agents. A person may be internal in the organization (e.g., administrator) or external (e.g., customer, attacker).
- Asset: a valuable resource, which can be a tangible asset (e.g., air-conditioning, fire extinguisher, computers) or an intangible asset. Intangible assets can be, for example, software, data, and industrial manufacturing processes.
- Location: Defines the asset's location. Location can be a brick and mortar physical location such as a classroom, data center or office. It can also consist of collaborative research materials on a file share or financial information stored in a database (Vogel, 2013).

b) Risk dimension

The concepts of the risk dimension are:

- Risk: a combination of a vulnerability and threat causing harm to one or more asset.
- Severity: the level of risk, e.g. high, medium or low.
- Threat: a violation of a security criterion. The threat may be natural, accidental, or intentional (attack).
- Vulnerability: a weakness of an asset or group of assets that can be exploited by one or more threats (ISO-b, 2004) (e.g., weak password).
- Impact: the impact may vary from a simple loss of availability to loss of the entire information system control. Impact can also be of other types such as harm to the image of the company.

- Attacker: the person (or program) who carries out the attack (intentional threat).
- Attack method: Refers to the different methods used by attackers to accomplish their attacks, such as sniffing (which lets attackers capture and analyze traffic transmitted over a network); spoofing (where the attacker attempts to impersonate someone or something else); and social engineering (tricking people into giving sensitive information or performing actions on behalf of the attacker).
- Attack tool: The tool used to perform the attack. e.g. sniffing tool (e.g., Wireshark[9]), spoofing tool (e.g. Subterfuge[10]), scan port tool (e.g., Nmap[11]) and others.

c) Treatment dimension

- Security goal: a security goal defines what a stakeholder hopes to achieve in the future in terms of security.
- Security Requirement: a condition defined on the environment that needs to be fulfilled in order to achieve a security goal and mitigate a risk. Depending on what we want to protect and on the target security level, we define our requirements. They can be related to databases, applications, systems, organizations, and external environments. For example, "the system shall ensure that personal data can be accessed only by authorized users" and "the system shall deliver data in a manner that prevents further or second hand use by unauthorized people".
- Control: a means or a way to secure assets and enable a security requirement, e.g., alarm or password.
- Security criterion: defines security properties such as confidentiality, integrity, availability, and traceability. It can also be considered as a constraint on assets.
- Requirements document: The document that states in writing the necessary security requirements to protect the assets. Two main documents generally contain security requirements:
- Security policy: a security policy expresses the defense strategy or strategic directions of the information security board of an organization.
- Specification document: it gathers the set of requirements to be satisfied by a material,

[9] http://www.wireshark.org/
[10] http://code.google.com/p/subterfuge/downloads/list
[11] http://nmap.org/

design, product, or service. The document contains, *inter-alia*, security requirements.

B. Relationships of the security ontology

High-level relationships between those concepts were defined. They were categorized into four kinds: *IsA, HasA, SubClassOf* and *AssociatedTo*. The relationships between the concepts of the security ontology can be briefly described as follows: An organization has assets (*Has_Asset*). An asset may have a location (*Has_Location*). Tangible and intangible assets are subclasses of the concept asset (*SubClassOf*). An organization also includes persons that it deals with (*Has_Person*). The persons can be internal or external (*SubClassOf*). An asset is threatened by one or many threats (*Threatens*). These threats exploit vulnerabilities in the assets (*Exploits*). The threat-agent leads an attack (*LeadBy*) and uses attack methods (*UseMethod*) or attack tools (*UseTool*) to achieve an attack. A person (internal or external to the organization) can (*TurnsInto*) a threat agent. A threat implies an impact (*Implies*), for example: "A denial of service attack implies a server downtime". The impact affects one or more assets (*Affect*). A threat can be natural, intentional, or accidental (*SubClassOf*). A threat generates a risk (*Generate*) with a certain level of severity (*HasSeverity*). Security requirements mitigate a risk (*Mitigate*) and satisfy (*Satisfies*) security goals expressed by stakeholders (*ExpressedBy*). Security requirements fulfill (*Fulfills*) one or more security criteria. For instance, the requirement "The application shall ensure that each user will be able to execute actions for which he/she has permission at any time/every week" satisfies the security criteria Confidentiality and Availability. Controls enable a security requirement (*Enables*). For example, the control "password" enables the requirement "The application shall ensure that each user will be able to execute actions for which he/she has permission". Security policies and specifications incorporate (*Includes*) security requirements, that can be either security software requirements (*SubClass*), that relate to the security of applications, databases, or security organizational requirements (*SubClass*), that relate to assets, persons, buildings.

C. Attributes and axioms of the security ontology

In addition to concepts and relationships, ontologies contains axioms and attributes. Formal axioms are assertions, accepted as true about abstractions of a field. The axioms allow us to define the meaning of concepts, put restrictions on the values of attributes, examine conformity of specified information or derives new ones (Staab and Maedche, 2000) . As stated before, the ontology proposed in this chapter was not created from scratch. It was

constructed by reusing knowledge of existing security ontologies. In particular, some attributes (see Table 5.1) of the ontology proposed by (Lekhchine, 2009) were reused. For instance, a person has a phone number (its type is string), a requirements document has a version (its type is string).

Table 5.1. Part of the table of attributes

Concept	Attribute	Value type
Person	Phone number	Integer
Software	Version	String
Requirement document	Version	String
Password	Minimum length	Varshar

The ontology proposed by (Fenz and Ekelhart, 2009) was a rich source of axioms. Table 5.2 illustrates some axioms with their descriptions and the related concepts.

Table 5.2. Part of table of axioms

Concepts	Description	Expression
Threat	A threat can be either intentional or accidental	$\forall\, x: Threat$ $\Rightarrow Intentional\ Threat\ (x)$ $\vee\ Natural\ Threat\ (x)$ $\vee\ Accidental\ Threat\ (x)$
Requirements document Security policy Specification	A requirements document can be either a policy or a specification	$\forall\, x: Requirements\ Document$ $\Rightarrow Security\ Policy\ (x)$ $\vee\ Specification\ (x)$

Figure 5.2 presents the security ontology proposed in this chapter. It includes the three dimensions, including concepts and relationships.

2 EVALUATION OF THE CORE SECURITY ONTOLOGY

Given that our goal was to develop an ontology covering the high-level concepts of security, and make it (re)usable by the requirements engineering community, the following criteria were the focus:

• *Completeness:* this criterion will be evaluated by mapping the target ontology and some other ontologies extracted from literature. The focus was mainly on security ontologies that have been used in security requirements engineering (Salinesi, Ivankina, and Angole, 2008) (Daramola et al., 2012) (Velasco et al., 2009).

• *Validity:* Through this criterion, the ability of the ontology to provide reliable answers to a set of questions using its terminology was checked.

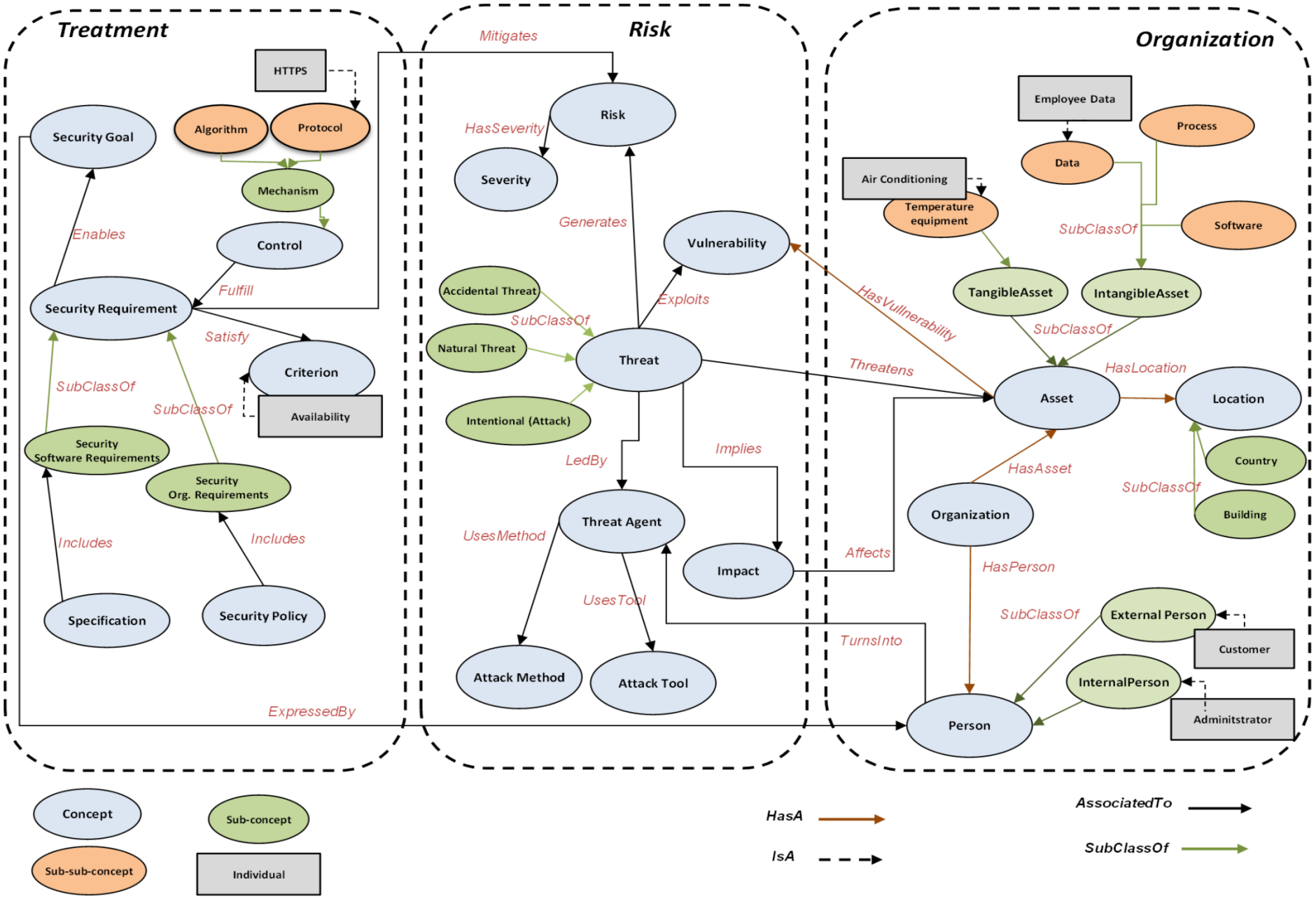


Figure 5.2. The core security ontology

• *Usability:* This criterion refers to the degree to which a person believes in using a particular system. In our case, it demonstrates that the ontology can be used for security requirements elicitation, and reused through different projects.

This chapter concentrates on the evaluation of completeness and validity of the security ontology. The evaluation of the usability criterion will be presented in chapter 10.

2.1. COMPLETENESS

The completeness criterion verifies that our ontology integrates the knowledge that exists in the other ontologies. An alignment table was drawn up, with on one side the concepts of our ontology. The other side contains concepts of security ontologies found in security requirements engineering literature (these particular ontologies were chosen, and not others, because they have been used in requirements engineering contexts). Table 5.3 presents the result of the alignment. .

Table 5.3. The alignment table of the proposed security ontology with ontologies used for security requirements elicitation.

	Ontologies used for security requirements elicitation				
Concepts of the ontology	(Daramola, Sindre, & Moser 2012)	(Salinesi, Ivankina, Angole 2008)	(Velasco et al. 2009)	(Salini & Kanmani 2012a)	(Dritsas et al. 2006)
Asset	Asset	Asset	Asset	Asset	Asset
Location	-	-	-	-	-
Organization	-	-	-	-	-
Person	-	-	-	Stakeholder	Stakeholder
Threat	Threat/ Active attack	-	Threat	Threat	Threat/ Deliberate attack
Vulnerability	-	Threat causes	-	Vulnerability	Vulnerability
Risk	-	-	Risk	-	-
Severity	-	-	Valuation criteria	-	-
Impact	-	-	-	Impact severity	-
Attacker	-	-	-	-	Attacker
Attack tool	-	-	-	-	-
Attack method	Code injection	-	-	-	-
Security goal	-	-	-	-	Objective
Security criterion	-	-	-	Security objective	Security requirement
Security requirement	-	Treatment	Security requirement	Security requirement	-
Control	-	-	Safe guard	-	Countermeasure

Most of the security ontologies used in SRE contain the concept of "Asset". Given that security issues affect all the infrastructure of organizations, other concepts were introduced (with their corresponding sub-classes): Location, Organization and Person. While many of the other security ontologies take into consideration the concept Threat, most of them neglect the concept Risk generated by a threat, and its Severity. Only the ontology proposed by Dritsas et al. (Dritsas et al., 2006) uses the concept of "Attacker". Only the ontology used by Daramola et al. (Daramola, Sindre, and Moser, 2012) includes the concept of "Attack Method". Our proposed security ontology covers the concept "Objective" used by Dritsas et al. (Dritsas et al., 2006). The concept "Security Criterion", missing in the security ontologies (Salinesi, Ivankina and Angole, 2008), (Daramola, Sindre, and Moser 2012) and (Velasco et al. 2009) was used in (Salini and Kanmani, 2012a) and (Dritsas et al. 2006). Note that (Dritsas et al., 2006) considers as a 'security requirement' what other sources consider a 'security criterion' (availability, confidentiality...). The concept "Security Requirement" was used in (Salinesi, Ivankina and Angole, 2008), (Velasco et al., 2009) and (Dritsas et al., 2006). These results tend to demonstrate that the proposed security ontology is complete with respect to the union of all the other security ontologies used in security requirements studies, since it incorporates all their concepts.

2.2. VALIDITY

According to Uschold and Gruninger (Uschold et al., 1996), informal and formal questions are one way to evaluate an ontology. The latter must be able to provide reliable answers to these questions using its terminology. This section lists a number of questions that a requirements engineer is likely to encounter during the requirements elicitation phase of a development project. These questions should be regarded as indicative of what the ontology can deal with and reason about. Table 5.4 summarizes some of these questions. Each of the questions is expressed informally in natural language and formally using SQWRL[12] (Semantic Query-Enhanced Web Rule Language) - an ontology query language. The answers to the questions are presented in the last column. These queries guide the requirements engineer during the security requirements elicitation process. The process includes: i) valuable assets identification (what are the assets of the organization?), ii) the risk analysis (what are the threats that threaten the asset? Who leads the attack? What is the attack method used?), and iii) security requirements elicitation (what are the security requirements to mitigate the risk?

[12] http://protege.cim3.net/cgi-bin/wiki.pl?SQWRL

What are the controls needed to implement those security requirements? What are the security criteria that those requirements fulfill?).

Table 5.4. Informal and formal questions to the ontology

	Queries	**Part of result**
Valuable asset identification	What are the organizations in the scope of the project?	Organization X, Organization Y, etc.
	Organization(?o) → sqwrl:select(?o)	
	What are the assets to be protected in the organization X?	IT systems, users' ids, user's password, keys, etc.
	Has_Asset(OrganizationX,?a) → sqwrl:select(?a)	
Risk analysis	What are threats that threaten the asset "IT systems"?	Unauthorized use of IT systems
	threatens(?T,IT systems) → sqwrl:select(?T)	
	Who is responsible of the threat "Unauthorized use of IT systems"?	Competitor
	LedBy(Unauthorized use of IT systems,?A)→sqwrl:select(?A)	
	What is the method used by the attacker 'competitor' to attack the IT systems?	Gain physical access
	Threat(Unauthorized use of IT systems)∧ Uses(competitor,?M) → sqwrl:select (?M)	
	What are the impacts of such a threat on the Organization X?	Theft of sensitive information
	Implies(Unauthorized use of IT systems,?I)→ sqwrl:select(?I)	
Security requirements elicitation	What are the security requirements to consider to mitigate the risk?	**Req1.** The organization should control access to its buildings and sensitive areas by using adequate control access procedures.
	Exploits (Unauthorized use of IT systems, V?) ∧ mitigated_by(?V, ?R) →sqwrl:select(?R)	**Req2**. After a defined period, inactive user account should be locked.

This section has demonstrated how the security ontology could be exploited in the security requirements elicitation phase. This can provide the requirements engineers with the necessary knowledge.

3 CONCLUSION

This chapter presented a core ontology for the IS security requirements elicitation and analysis process. The completeness of the ontology proposed in this chapter was evaluated

with regards to existing security ontologies used in security requirements engineering methods. The core security ontology was implemented as part of the tool AMAN-DATool (Appendix D) and was evaluated during a controlled experiment with end users (Chapter 10).

Despite our effort, the goal of constructing this kind of security ontologies remains ambitious and is was found to be more complex than expected. One single team's work is not large enough. This research can be improved through more collaboration with teams that worked on security ontologies. The ontology is of a generic nature; more mechanisms need to be considered to use it for different domains.

In the next chapters, and in the context of AMAN-DA, we integrated the ontology and its reasoning features with Secure Tropos – the security requirements analysis approach. We will present how to make this security ontology more domain specific by relying on domain ontologies.

Chapter 6

Security goals and security requirements

Security goals and security requirements are both two cornerstones concepts in our proposition. As illustrated in the Figure 4.1. (cf. chapter 4), stakeholders' security goals are inputs of the AMAN-DA method, security requirements are part of the security ontology and are outputs of the method. Stakeholders can express security concerns at different levels of detail. Therefore, there is a distinction between security goals (abstract) and security requirements (more detailed). According to (Fabian et al., 2010), the distinction between security goals and security requirements is not readily established in the requirements engineering community; it is not completely precise due to vagueness of subjective intuitions and semantic intricacies of natural languages. In this chapter, we raise the question: *What would be the (linguistic) pattern to specify security goals and security requirements?*

AMAN-DA distinguishes between security goals and security requirements. In this chapter this distinction is explained. Two syntactic models are proposed: (i) a security goal model to capture stakeholders' goal, and (ii) a syntactic security requirements pattern to specify security requirements that materialize the security goals.

1 Between Security goals and security requirements

Many authors implicitly assume that security requirements are similar to high-level security goals. (Tettero et al., 1997) are explicit about this, defining security requirements as the confidentiality, integrity, and availability of the entity for which protection is needed. While this is a clear definition, in some cases it may not result in precise enough requirements. Consider an example in health care: both doctors and administrators would probably agree on the importance of confidentiality, integrity, and availability of the clinical information, but they could disagree on the concrete security requirements that express those goals. The requirements need to be more explicit about *who* can do *what* and *when*.

Some authors identify security requirements with security policies. (Devanbu and Stubblebine, 2000) define a security requirement as "*a manifestation of a high-level organizational policy into the detailed requirements of a specific system. [...We] loosely (ab)use the term 'security policy' [...] to refer to both 'policy' and 'requirement*'". Anderson (Anderson, 2001) is less direct; he states that a security policy is "*a document that expresses [...] what [...] protection mechanisms are to achieve*" and that "*the process of developing a security policy [...] is the process of requirements engineering*". (Haley et al., 2006) consider that the difficulty with security policies is their chameleon-like meaning. The term can be used for anything from a high-level aspiration to an implementation. Therefore, without accompanying detailed explanation, it is not satisfactory to define security requirements as security policies. However, security policies may contain security requirements. (Lee, Lee, and Lee, 2002) point out the importance of considering security requirements in the development life cycle, but do not define them.

A number of papers (some of them presented in the state of the art of the thesis) focused on security requirements by describing how they may be violated. For example, (McDermott and Fox, 1999), followed by (Sindre and Opdahl, 2000) who described abuse and misuse cases, extending the use case paradigm to undesired behavior. In Secure i*; (Liu, Yu, and Mylopoulos, 2003) describe a method of analyzing possible illicit use of a system, but omit the important initial step of identifying the security requirements of the system before attempting to identify their violations.

This quick bibliographic tour indicates that the difference between security requirements and security goals needs to be more precisely defined. This is the objective of the next sections.

2 SECURITY GOALS

2.1. THE NOTION OF GOAL

Goals have been recognized as an effective way to identify requirements for a long time now (Potts, 1997) (Rolland and Salinesi, 2005). They are known to be essential components involved in requirements engineering process (Van Lamsweerde, 2001).

Goals are the objectives and targets of achievement for a system. According to (Van Lamsweerde, 2001) : "*A goal is an objective the system under consideration should achieve.*"

Goals are expressions of intent and thus declarative with a *perspective* nature, by opposition to descriptive statements (Rolland and Salinesi, 2005).

According to Jackson (Jackson, 1995), a goal is an "optative" statement, expressing a state that should be achieved or maintained. In other words, the goal represents the objective that we want to achieve ("What?") without saying "How?" to execute it.

Goals may be formulated at different levels of abstraction, ranging from high-level concerns (such as "provide ubiquitous cash service" for an ATM network system) to low-level, technical concerns (such as "acceleration command delivered on time" for a train transportation). The system which a goal refers to may be the current one or the system-to-be; both of them are involved in the requirements engineering (RE) process.

In requirements engineering, goal-driven approaches (KAOS (Van Lamsweerde, 2007), i* (Yu, Strohmaier, and Deng, 2006), and Tropos (Bresciani et al., 2004)) focus on why systems are constructed, expressing the rationale and justification for the proposed system. Focusing on goals, instead of specific requirements, allows analysts to communicate with stakeholders using a language based on concepts with which they are both comfortable and familiar (Antón and Earp, 2000). Goals are also known to be a beneficial source for requirements derivation. In fact, goals are operationalized and refined into requirements.

Moreover, goals provide a precise criterion for sufficient completeness of a requirements specification; the specification is complete with respect to a set of goals if all the goals can be proved to be achieved from the specification and the properties known about the domain considered. Goals have various other roles in RE, such as the exploration of design choices, requirements tractability, requirements negotiation, conflicts detection and resolution, etc (Fabian et al., 2010).

Goals must obviously be specified precisely to support requirements elaboration, verification/ validation, conflict management, negotiation, explanation and evolution. An informal (but precise) specification should always be given to make it precise what the goal name designates (Tettero et al., 1997).

With the emergence of goals to the field of requirements engineering, (Prat, 1997) proposed a model to this notion. This model is derived from a linguistic approach, inspired by case grammar of Fillmore (Fillmore, 1967), and extensions from Dik (Dik, 1997).

According to the model illustrated in Figure 6.1., a goal is described under the form of a verb associated to a target and completed by a set of optional parameters. Stating a goal is then represented by a verb, targets and different parameters which play specific roles in relation to this verb.

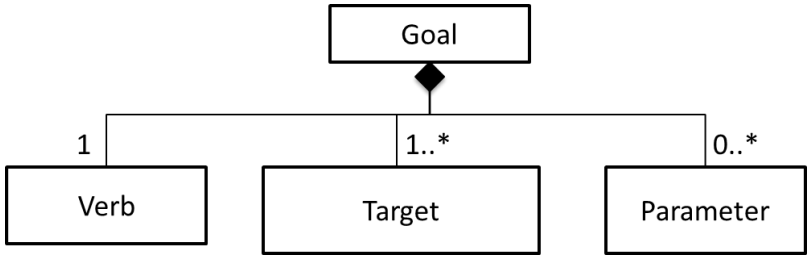


Figure 6.1. Part of the goal model proposed by Prat (Prat 1997)

In this model, the verb sets the action for the realization of the goal, while the target is either the existing object before performing the intention or the result arising from the satisfaction of the goal. The parameters (ways, direction, quality, quantity and beneficiary) can be useful to clarify the goal or express additional information. For example, G1= {#prepare, #proposition, ∅} is composed of the verb "prepare" and the target "proposition", that represent the actual result of the satisfaction of the goal.

Last, recall that goals cover different types of concerns: functional concerns associated with the services to be provided, and non-functional concerns associated with quality of service -- such as safety, **security**, accuracy, performance, and so forth. Security goals will be described in the next section.

2.2. SPECIFYING SECURITY GOALS

In the mid ninetieth, many research explored the notion of goals to analyze security at the requirements engineering early phases. The state of the art in the beginning of the thesis report (Chapter 2) presents some goal-oriented approaches for security requirements engineering. All of these approaches are model-based. Their idea of security goals differ slightly from an approach to another as well as the concrete syntax they used to model this concept.

- With Secure i*, (Liu, Yu, and Mylopoulos, 2003) identify security requirements during the development of multi-agent systems by analyzing the relationships between strategic actors, such as users and stakeholders, and potential attackers. Authors use the concept of **soft-goal**

to model and reason about security requirement. Soft-goals in Secure i* are graphically represented into a cloud. Iintegrity, confidentiality, availability are examples of softgoals.

- Secure Tropos (Mouratidis and Giorgini, 2007) (that was based on Secure i* and Tropos methodology) use the same concept of **soft-goal;** it also uses the concept of **security constraint.** A security constraint is restriction related to security issues, such as privacy, integrity and availability, which can influence the analysis and design of a multi-agent system under development by restricting some alternative design solutions, by conflicting with some of the requirements of the system, or by refining some of the system's objectives. They are graphically represented as hexagons that are labeled with a constraint. An example of a security constraint would be "Keep patient anonymity".

- With KAOS, Lamsweerde (Van Lamsweerde, 2007) relies on the notion of anti models and **anti-goals**. An anti-model is constructed after the goals of the system-to-be have been elaborated and refined. Anti-models represent the scenario unwanted to happen. Anti-goals are obtained by negating existing goals. Once an anti-model stands and the resulting obstacles have been identified, the requirements engineers are expected to develop countermeasures so that the preconditions of the anti-goals are no longer fulfilled.

Browsing the literature indicates some other definitions related to security goals. Security goals define what a stakeholder hopes to achieve in the future in terms of security. According to Fabian et al. (Fabian et al., 2010):

"*A stakeholder's security goal expresses his or her security concerns towards an asset*"

"*Security goals are defined as very general statements about the security of an asset"*

"*Security goals are traditionally classified into integrity, confidentiality, and availability goals.*"

While according to ISO/IEC 13335-1:2004 (ISO-b, 2004):

"*Integrity* is the property of safeguarding the accuracy and completeness of assets."

"*Confidentiality* is the property that information is not made available or disclosed to unauthorized individuals, entities, or processes."

"*Availability* is the property of being accessible and usable upon demand by an authorized entity."

Based on what we found in the literature and based on the general goal model proposed by Prat (Prat, 1997), we propose a linguistic model to specify security goals (as presented in Figure 6.2.):

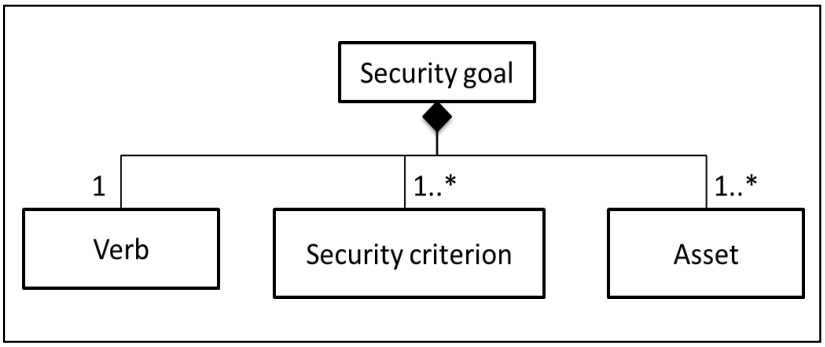


Figure 6.2. Security goal model.

This can be read as follows:

Security goals are expressed as a clause with (i) a main verb, (ii) one or many security criteria (e.g. confidentiality, integrity, availability) and (iii) target asset (s) that need to be protected.

Thus:

Security goal = <Verb> <Security criterion> <Assets>.

For example, the customers of a bank may have the goal that their financial situation remains confidential. This can be formalized as "(Maintain) $_{\text{Verb}}$ (the confidentiality) $_{\text{Security criterion}}$ of (financial situation) $_{\text{Asset}}$"

The public, represented by a government agency, may have the goal to maintain the integrity of electronic financial transactions.

(Maintain)$_{\text{Verb}}$ (the integrity) $_{\text{Security criterion}}$ of (electronic financial transactions) $_{\text{Asset}}$

Security goals are materialized by security requirements, as the next section will describe it.

3 SECURITY REQUIREMENTS

Security requirements capture security goals in more detail. A security requirement refines one or more security goal. Mayer (Mayer, 2012) defines a security requirement as "*a condition defined on the phenomena of the environment that we wish to make true by installing the IS (Information System) in order to mitigate risks*". He added that each security requirement contributes to cover one or more risk treatments for the target IS. (Fabian et al., 2010) consider that a security requirement refer to a particular piece of information or service that explicates the meaning of the asset it concretizes in the context of the system under construction.

A security requirement also indicates the counter-stakeholder against whom the requirement is directed. This is particularly important for confidentiality requirements, where the counter-stakeholder is the party who must not get to know the information to which the requirement refers. A counter-stakeholder is not necessarily an adversary who tries to attack the system.

Another important aspect of a security requirement concerns the circumstances in which it must be satisfied. These describe application conditions of functionality, temporal, or spatial aspects, the social relationships between stakeholders— in general, the ‘‘context’’ to which the requirement refers.

(Haley et al., 2006) define security requirements as *constraints on the functions of the system, where these constraints operationalize one or more security goals*. Authors consider that security requirements should express the system's security goals in operational terms, precise enough to be given to a designer/architect. Security requirements, like functional requirements, are prescriptive, providing a *specification* (behavior in terms of phenomena) to achieve the desired effect.

3.1. SPECIFYING SECURITY REQUIREMENTS (SYNTACTIC PATTERNS)

Wiegers (Wiegers, 2003) recommend to write individual requirements, and to avoid long narrative paragraphs that contain multiple requirements. Authors warn us that readers shouldn’t have to glean the individual requirements embedded in a mass of free-flowing descriptive language. It is advised to distinguish individual requirements from background or contextual information. These recommendations should be applicable also to security requirements known to be "*the most difficult of requirements types",* and potentially the ones

causing the greatest risk if they are not correct (Robertson and Robertson, 2013). This motivates the need for a comprehensive reusable patern to specify security requirements.

Klaus Pohl (Pohl, 2010) use the notion of *syntactic requirements patterns* for documenting requirements. The use of syntactic patterns aims, right from the beginning, at avoiding mistakes that frequently occur when defining textual requirements. An example of a frequent mistake is the use of the passive voice. Syntactic requirements patterns are defined based on experience with syntactic structures for textual requirements and are applied to support the documentation of requirements. The term "syntactic requirement pattern" is defined as follows:

A syntactic requirements pattern defines a syntactic structure for documenting requirements in natural language and defines the meaning of each part of the syntactic structure.

(Pohl, 2010) reports an example of a syntactic pattern for documenting requirements in natural language proposed by (Rupp, Simon, and Hocker, 2009).

The pattern is as follows:

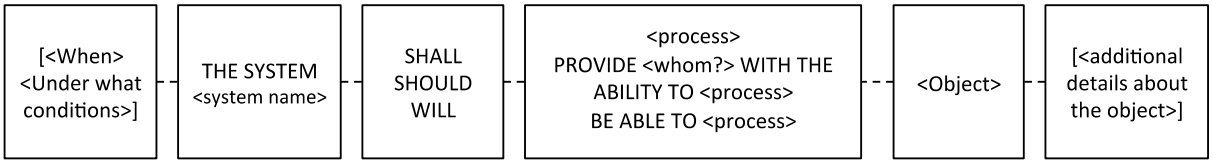


Figure 6.3. Syntactic requireements pattern for documenting requirements with a condition (Rupp, Simon, and Hocker 2009)

As Figure 6.3 depicts it, the pattern is composed of 6 main elements. The first element defines the time (when) or conditions under which the function of the requirement shall be performed. The second one defines the name of the system, which shall provide the documented function. The modal verbs "*Shall/Should/Will*" indicate the importance of the requirement. For instance, the modal verb "Shall" indicates a legally binding requirement, while "Should" indicates highly recommended but would not make the system unacceptable if it is not implemented. The *Process* indicates the required functionality. This functionality is documented by a full verb such as "print" or "transfer". The *Object* describes the object for which the functionality is required, such as the type of document to be prented (e.g. tax form). The object as well as *additional details* about the object are documented after the process.

Based on the pattern of (Rupp, Simon, and Hocker, 2009), we propose the following syntactic pattern for documenting security requirements (Figure 6.4):

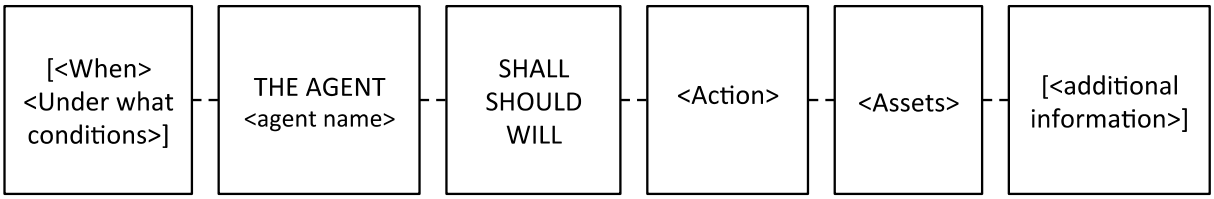


Figure 6.4. A syntactic requirements pattern for documenting security requirements.

This pattern consists of the following main structural elements:

- *<When>:* This element defines the temporal or the logical conditions under which the function documented in the requirement shall be performed.
- *<Agent name>:* This element defines the name of the agent, which shall provide the documented function. As security requirements are defined depending on what we want to protect and on the target security level, they can be related to databases, applications, systems, organizations, and external environments. That is why the *agent* who performs the requirement can be a system (e.g. "The system should lock accounts"). It can also be any *agent* of the domain. (e.g. "The chief engineer should provide fire extinguishers"). This agent is the grammatical subject of the sentence.
- *"Shall/Should/Will":* Same as in the pattern of Rupp (Rupp, Simon, and Hocker, 2009) These elements indicate the importance of the requirement. They are consistent with the overarching objective of clear and effective communication.
- *<Action>:* Actions constitute the activity or set of activities that processes the requirement (e.g. provide, lock, etc.)
- *<Assets>:* Assets are the objects used for, or are part of the actions (e.g. fire extinguishers, accounts). They can also refer to controls as presented in the security ontology (Chapter 5).

In the banking example (cf. Section 2.2), the security goal expressed by customers as: "(Maintain) $_{\text{Verb}}$ (the confidentiality) $_{\text{Security criterion}}$ of (financial situation) $_{\text{Asset}}$"

Can be materialized by the security requirement:

(The system) $_{\text{agent}}$ (should) (ensure that) $_{\text{Action}}$ (customer's balance of account) $_{\text{assets}}$ (is hidden from arbitrary bank employees) $_{\text{additional information}}$

In this example, the agent is the system under construction; the asset is the customer's balance

of account.

3.2. SECURITY REQUIREMENTS IN THE CORE SECURITY ONTOLOGY

In chapter 5, the core security ontology was presented. This security ontology encapsulates and formalizes the security related knowledge. As reported before, in addition to threats, vulnerabilities, security criteria, attack methods, etc, the security ontology contains a set of relevant security requirements.

In fact, security requirements in the core security ontology are organized according to the syntactic security requirements pattern presented in this chapter. This organization, allows making the security requirements in the security ontology well formalized and reusable when the security ontology is used in different domains.

Figure 6.5. illustrates a deeper view the core security ontology.

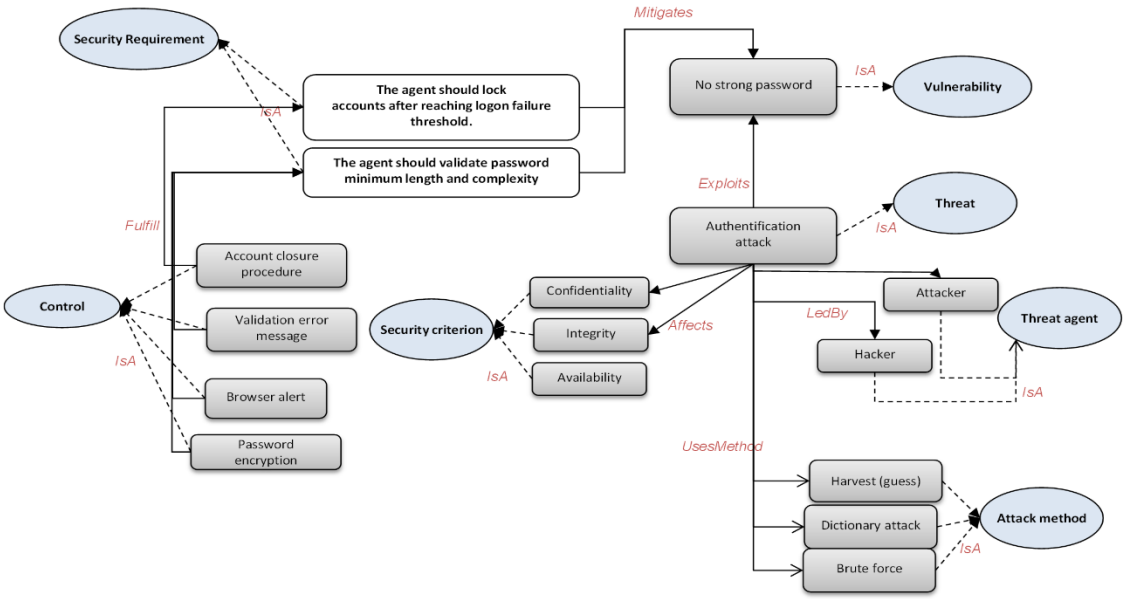


Figure 6.5. Deeper view on the core security ontology presented in chapter 5.

4 CONCLUSION

Since AMAN-DA is meant to be a generic method to be applied in different domains of application, well-formed artifacts were needed. In this chapter, the concentration was on security goals and security requirements. Based on previous well-referenced models in literature, a model to specify security goals was proposed. A reusable synthetic pattern to document security requirements was presented. The next chapter will present another facet of AMAN-DA, the domain knowledge.

Chapter 7

Domain knowledge and domain ontologies

AMAN-DA, the proposed method aims at providing not only the necessary security related knowledge for security requirements elicitation, but also at providing specific domain knowledge for the same purpose. This is made possible thanks to domain ontologies. Domain ontologies can provide knowledge about the domain assets that need to be protected, the agents of the organizations, their roles and responsibilities, organizations in the scope of the study, the organizational activities. Such a knowledge is necessary to perform a security requirements elicitation process. Knowing the assets helps to detect the ones that are more subject to vulnerabilities and sensitive to threats but also the ones that can be used as controls to mitigate those threats. Knowing the agents in the organization and their roles helps in identifying the potential threat agents (attackers). In addition to that, having the domain knowledge at hand helps while interviewing stakeholders and identifying their security goals. The other goal AMAN-DA wants to achieve is to be able to be used for different domains, and not to stick to only one single domain (one single domain ontology).

To address this issue, this chapter tackles the research question: "*How to represent the domain knowledge in a generic way to allow AMAN-DA to switch from one domain to another?"*

This chapter will present the formal representation of the domain knowledge. It introduces and describes a multi-level domain ontology. The latter contains an upper level that covers high level concepts and relations that are shared with all domains. It also contains a lower level that contains the information specific to the application domain which can vary from a domain to another.

Section 1 in this chapter describes the multi-level domain ontology. Sections 2 and 3 propose the representation of the online shopping and the maritime ontologies (domains).

1 DOMAIN ONTOLOGIES: FORMALLY REPRESENTING THE APPLICATION DOMAIN KNOWLEDGE

To achieve more efficient supports for security requirements elicitation of higher quality, AMAN-DA method utilizes domain knowledge. In this research trend, one of the major issues is which artifact to model and how to represent domain knowledge for security requirements elicitation. The approach as discussed before uses a domain ontology as a representation of the domain knowledge.

Ontology technologies are frequently applied to many problem domains nowadays (Grüninger and Lee, 2002). Since concepts, relationships and their categorizations in a real world can be represented with ontologies, they can be used as resources of domain knowledge.

An ontology is considered (Maedche and Staab, 2001) as a thesaurus of words and inference rules on it, where the words in the thesaurus represent concepts and the inference rules operate on the relationships on the words. A domain ontology also defines the kinds of things that exist in an application domain; it is the principal conceptual and descriptive structure for capturing the elements that exist in a domain. It is generally comprised of domain concepts (or classes), coherently linked together in a logic structure that, when complete, forms a graph. The domain ontology should represent the application domain knowledge and business information required for building software applications in a specific domain. It also includes the semantic relationships established among their concepts from a real-world point of view (Castañeda et al., 2010).

There are major benefits of applying ontology as a domain knowledge. Defining rich domain ontologies allow the reuse of knowledge in a domain. Domain ontologies have the advantage of being extensible, allowing the use of the basic concepts of the ontology and extending it according to the application domain where it is used. Finally, a very useful advantage lies in the fact that the explicit representation of the domain concepts in domain ontologies allows reasoning on the whole set of concepts.

For all these reasons, in addition to the role of domain in security requirements engineering, the proposed method, AMAN-DA, relies on domain ontologies for security requirements engineering.

1.1. MULTI-LEVEL DOMAIN ONTOLOGY

In the literature, we find a number of domain ontologies. As an illustration, we mention (Boyce et Pahl, 2007) who presented ontologies for the areas of databases and enzymology; (Mohan and Arumugam, 2005) which developed an ontology for the railways industry domain; and (Bjørner, 2007) which describes an ontology for the container line industry domain. The DAML Ontology Library proposes many domain ontologies in a wide variety of domains that researchers and developers in knowledge engineering communities, in particular the Semantic Web community, are constructing. Most of them are represented with standardized OWL based language so as to be exchanged by many people.

To allow AMAN-DA to be independent of any pre-selected domain ontology – i.e. to allow the method to be applied to any domain, also to provide an extensible ontology well structured, easy to understand, to integrate and to extend, we developed a multi-level domain ontology. The "*Multi-level Domain Ontology*", whose main concepts are represented in Figure 7.1, relies on previous studies on domain ontologies (Bjørner, 2010a), (Bjørner, 2010b), (Kaiya and Saeki, 2006), (Borgol et al., 2009), (Bittner, Donnelly, and Smith, 2004), (Wang et al., 2004). The ontology was designed to be easily extensible by sub-ontologies, depending on the domain at hand and depending on the system it is applied to.

The Multi-level domain ontology is represented as follows:

- ***Upper-domain ontology (upper level):*** The upper domain ontology represents a domain according to two main components: (a) *concepts* to which a domain refers (such as vessel, car, person, patient, etc.) and (b) *relationships* between those concepts (*HasLength*, *HasLocation*, etc.). All domains share common concepts represented by this upper view. Concepts can be *perdurants* or *endurants*. *Endurant* concepts are those entities that can be observed – perceived as a full, at no matter which given snapshot of time. Examples include agents and material objects (such as an apple). *Perdurants* are entities that unfold themselves over time in successive temporal parts or phases. *Perdurants* include behaviors (vessel voyage, visiting ports, etc.), events (e.g. new employee arrival) and actions (e.g. remove container from vessel, develop web page).

  Relationship between those concepts include: *Is-A* relationships (*generalization/specialization*, *HasA* (aggregation) (e.g. haveParents), *SubClassOf* (e.g. a registered customer is a *SubClass* of web customer), *AssociatedTo* (e.g. the

class ‘Web developer’ is associated with the relationship *PerformAction* to the class ‘Develop web application’).

- ***Specific domain ontology (lower level):*** contains the specific information to the application domain. The details of basic domain information (concepts and relations) represented in the upper ontology are clearly defined at this lower level which is specific to a particular domain and can vary from a domain to another. For example the concept ‘agent’ which is an endurant concept at the upper level can be specialized into ‘customer’ in a business domain ontology, the same concept can be ‘patient’ in a health domain ontology or a ’teacher’ in a education-school domain ontology.

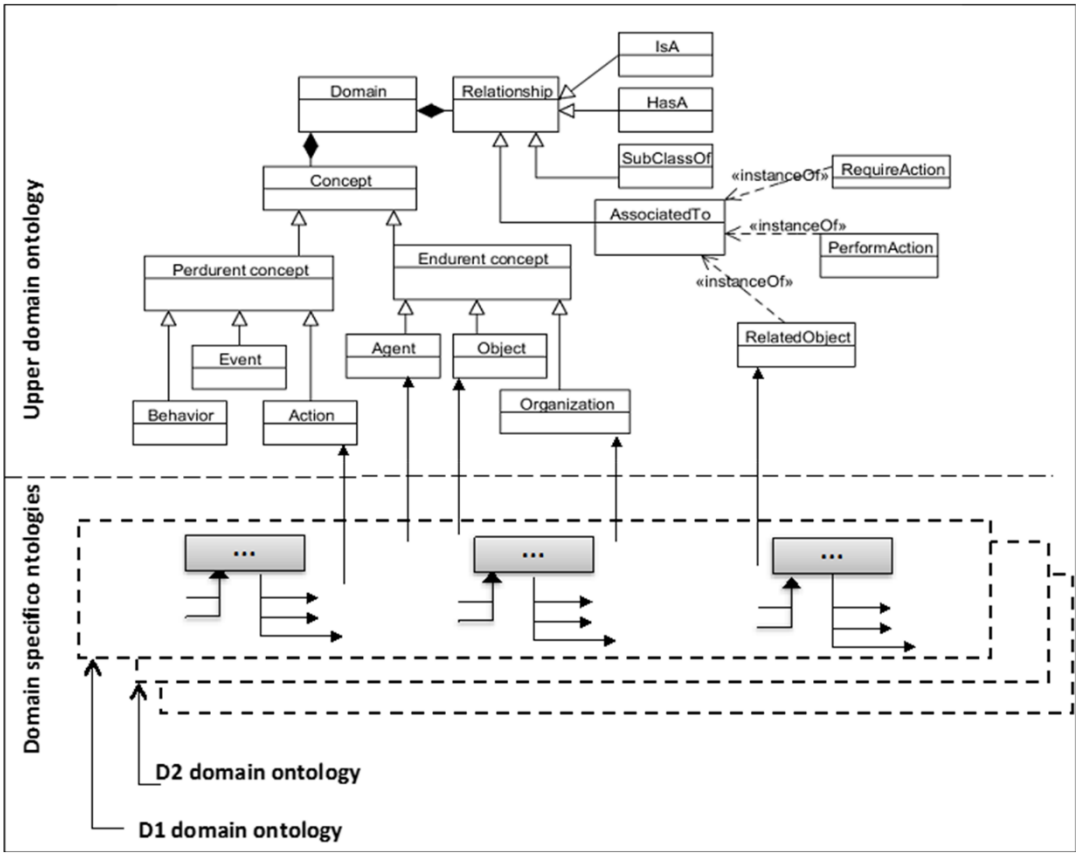


Figure 7.1. Multi-level domain ontology.

Based on the upper domain ontology, we distinguish key concepts and relations that AMAN-DA method relies on:

Organization: An organization is the structure that includes the different resources (human, hardware, and software). This concept provides all the organizations related to a domain of application. In a health domain ontology for example, we can find the ‘hospital’, and ‘the insurance company’ as two distinctive organizations of the domain, instances of the concept

‘Organization’.

Object: The concept ‘object’ refers to the material resources of an organization in a particular domain of application. This domain information is used to specify the ‘assets’ field in security goals expressed by stakeholders. It can also used to specify the ‘asset’ field in the security requirements that are stored in the security ontology.

Agent: The agent is an important domain information. An agent of the domain can be designated to be the person responsible for the security requirement. It can also be the threat agent who carries out an attack.

Action: An action is “something an agent does” (Bjørner, 2010a) . This perdurant concept covers the different actions that are carried in an organization by its different agents in a particular domain of application.

Two main relations associate the ‘agents’ with the ‘actions’: “*PerformAction*” and “*RequireAction*”, both of them are *AssociatedTo* relations. *RequireAction* defines the action required by an agent. For example, in maritime domain ontology, the agent ‘captain’ requires the action ‘maintain the ship engine’. *PerformAction* defines the agent responsible for performing a particular action. Following the example, the agent ‘the chief engineer’ performs the action ‘maintain the ship engine’.

The relation ‘*RelatedObject’* associates an ‘action’ to one or more ‘object’. This is to express that actions can be in relation with objects. For example, the action ‘calculate the student pass mark’ will be related to the object ‘student record’.

This multi-level organization of domain ontologies, with an upper level that contains the main, generic and shared concepts of all domains, and a lower level containing a particular domain ontology, the defined concepts and relations, allow us to use the domain knowledge needed for the security requirements engineering process undertaken by our method.

2 THE ONLINE SHOPPING EXAMPLE (DOMAIN ONTOLOGY)

Figure 7.2. illustrates part of an online shopping domain ontology. This ontology encapsulates the knowledge related to online shopping. In this ontology, one can see the organization (Online shopping company); its agents (web developer, web applications, web-customer, registered customer, new customer); the different actions, between these agents (e.g. register web-customer, show items, make purchase); and also the different objects related to these

actions (e.g. customer name, customer password, items). Each concept in the lower level domain ontology is labeled in blue by its corresponding concept in the upper level domain ontology.

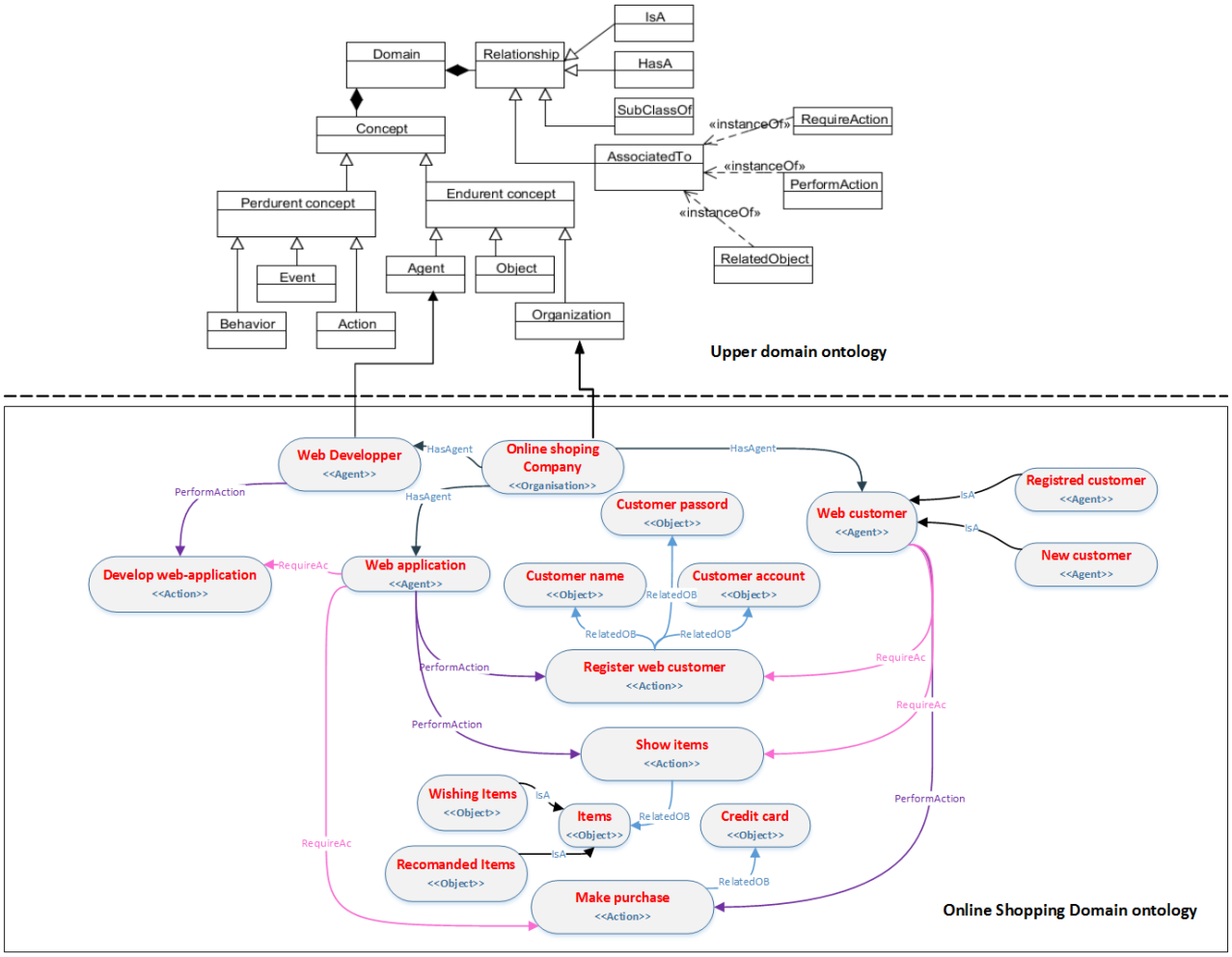


Figure 7.2. Online shopping domain ontology.

This domain ontology will be used to illustrate the use of domain knowledge in the security requirements elicitation method (cf. Chapter 8).

3 THE MARITIME DOMAIN ONTOLOGY

Applying the same principle described in this chapter (multi-level domain ontology), a maritime domain ontology was developed. This domain ontology was the one used during the evaluation of AMAN-DA as will be reported in the chapter of the case study (Chapter 9)

The maritime domain ontology contains 85 concepts and 115 relations. Figure 7.3. in the next page gives an overview of part of the ontology. This maritime ontology presents the agents of the maritime domain (e.g. Captain, Chief engineer, Duty officer, Crew, etc.). It also presents some actions that these agents perform or require (e.g. manage loading/unloading of goods, maintain the ship engine, manage weather and data equipment, manage ship documents, etc.). The ontology presents many objects related to the actions such as (water flood sensor, abandon ship alarm, engine plans, ocean current map, etc.).

4 CONCLUSION

In this chapter we have proposed a formal presentation of the domain knowledge through the multi-level domain ontology, and explained its main concepts and relations. The next chapter will present *how* the different mechanisms, including the domain ontology, will be used by AMAN-DA method to assist the requirements engineer in the elicitation of domain specific security requirements.

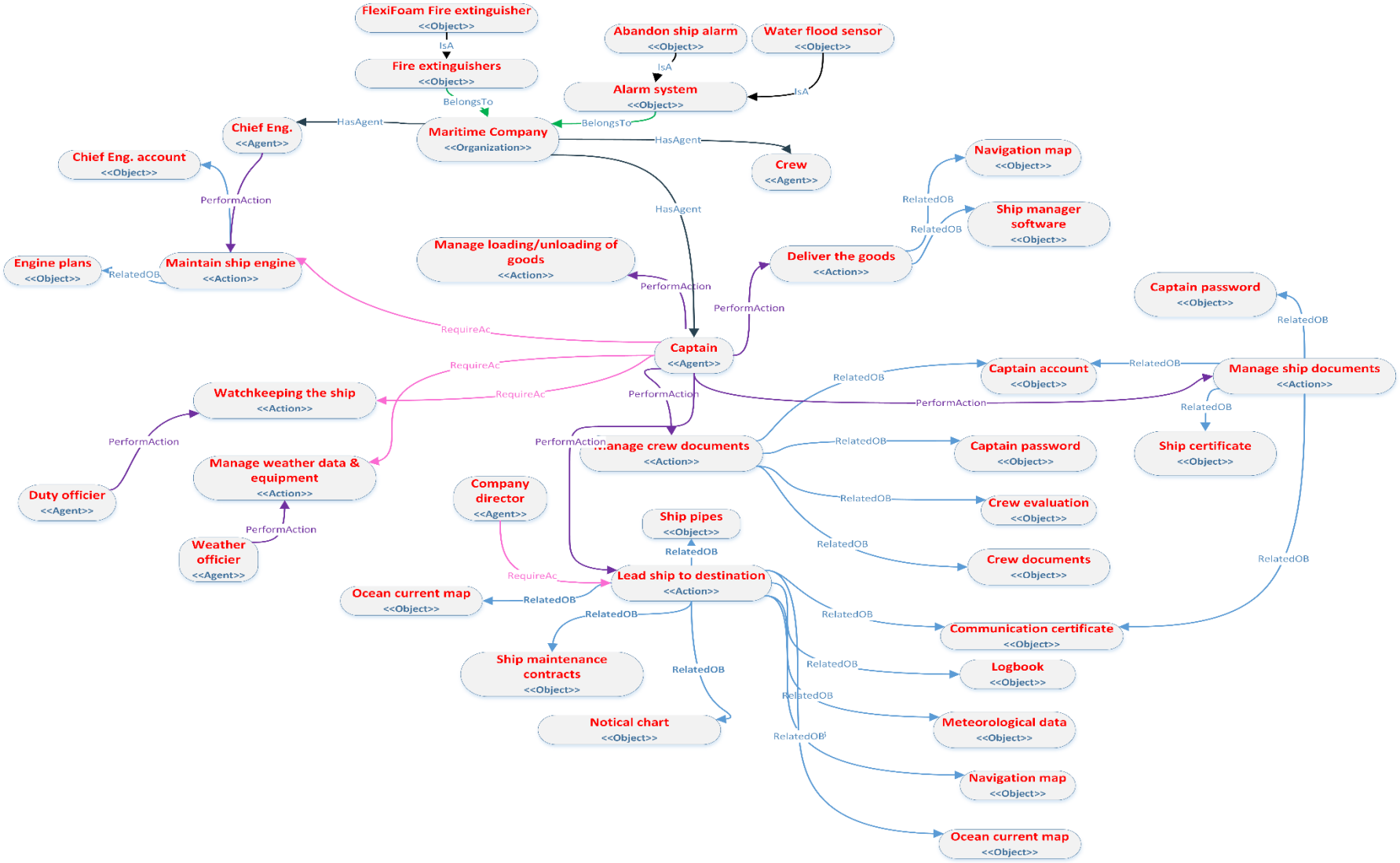


Figure 7.3. Part of the maritime domain ontology

Chapter 8

Ontology based domain security requirements elicitation

AMAN-DA, the method proposed in this PhD thesis, is based on three main elements. Chapters 4, 5, and 6 presented these elements (core security ontology, security goals and requirements and the multi-level domain ontology). The question that remain open is: "*how to put "everything together" to perform domain specific security requirements elicitation and analysis?"* This chapter answers this question by proposing the necessary algorithms and rules to achieve that goal. This will be illustrated through the online shopping running example presented in chapter 4.

1 DOMAIN SPECIFIC KNOWLEDGE DISCOVERY ALGORITHM

Let us come back to the running example presented in chapter 4:

Zach prepares specifications for different products. In these specifications, he usually focuses on functional requirements. His tacit knowledge about security and his primitive knowledge about the domain for which he elicits security requirements make his resulting security requirements poor and too generic. In addition, an interview of five minutes with stakeholders is not enough to know all the domain agents and objects that are present in the project. Sometimes, Zach copy/pastes the security requirements from other projects in an ad-hoc way, which is relatively time consuming and disorganized way of working.

This time, Zach is asked to elicit security requirements for a web application of an online shopping company. The engineer wants to i) capture the stakeholders' goals in a quick manner, ii) produce a well-formed Secure Tropos model, iii) generate the corresponding well–formed textual security requirements and iv) throughout the elicitation process, he wants to get access to domain knowledge related to online shopping without having to refer to the stakeholder.

One of the security goals captured by Zach after interviewing the stakeholder is the necessity to maintain the confidentiality of users' information.

This goal is formalized as recommended in chapter 6 as:

"(Maintain) $_{\text{Verb}}$ (the confidentiality) $_{\text{Security criterion}}$ of (user's information) $_{\text{Asset}}$".

One of the security requirements (in the core security ontology) that materialize this goal is "The agent should lock the account after reaching the logon threshold".

One can notice that the "user's information", "the agent", and "account" are generic information. We need a mechanism that makes the security related; generic knowledge expressed more domain specific.

The procedure presented below computes the semantic relatedness between concepts of the security ontology, concepts of the captured goals, and concepts of the domain specific ontology (e.g. the online shopping ontology in chapter 7). The goal is to select the most appropriate concepts in the domain ontology to replace the generic ones that exist in the security ontology (security requirements) and the security goals. Note that this procedure was inspired and adapted from other works in the field of web semantics (Paolucci et al., 2002), (Toninelli, Corradi, and Montanari, 2008), (Najar et al., 2012).

```
1.  Function EquivalentDomainConcept (Concept C_generic, Type-Concept TC, Ontology DomainOnt)
2.  C_score = 0;
3.  ConceptThresholdList= 0;
4.  For each C_domain in DomainOnt do
5.    If Type-Concept C_generic = TC do
6.      ConceptList += C_generic;
7.      C_score = Sim (C_generic, C_domain)
8.     If C_score = 1 do
9.        return ConceptList [C_generic];
10.    end if
11.    If C_Score >= t  do
12.      ConceptThresholdList. Add ( C_domain);
13.    end if
14.   end if
15. endfor
16. If ConceptThresholdList <> 0
17.   return Sort (ConceptThresholdList);
18. else
19.   return (ConceptList);
20. endFunction;
```

Figure 8.1. The domain specific knowledge discovery algorithm

Figure 8.1. presents the domain specific knowledge discovery algorithm. It takes as an input a generic concept ($C_{generic}$), its type (TC) (e.g. agent, object, organization), and a domain ontology (DomainOnt). For all existing concepts (C_{domain}) in the domain ontology, if the type of the concept (C_{domain}) corresponds to the type of the concept ($C_{generic}$) in the input (Type-Concept $C_{generic}$ = TC), the function Sim ($C_{generic}$, C_{domain}) (line 7) calculates the degree of semantic similarity (C_{score}) between the concept ($C_{generic}$) and (C_{domain}). As mentioned before, $C_{generic}$ corresponds to concepts in the security ontology or security goals. C_{domain} corresponds to concepts in the domain ontology.

The function Sim ($C_{generic}$, C_{domain}) uses edit distance and other metrics based on words' semantics in order to determine the degree of similarity of two concepts. As we are exclusively using the concepts' names, we selected the Wu & Palmer (Wu et Palmer, 1994) metric that is based on WordNet's structure. WordNet (Miller, 1995) is a structured English dictionary which provides specialization and composition relations between word senses. Wu and Palmer described a metric based on the depth of the words and their lowest common ancestor in the tree formed by composition relations.

The function Sim ($C_{generic}$, C_{domain}) returns a score between 0 and 1. In our case, if the C_{score} equals 1 (line 8) that means there is a domain concept that corresponds exactly to the generic concept ($C_{generic}$) (e.g. $user_{generic}$/$user_{domain}$). In this case, there is no need to replace it by the domain concept, and the algorithm returns the list ConceptList [$user_{generic}$].

If C_{score} is between 0 and 1 ($0 < C_{score} < 1$), we set a threshold (t), and the algorithm (lines 11, 16, 17) returns a sorted list of domain concepts (ConceptThresholdList) whose similarity is greater than t.

For example, table 8.1 displays the results of similarity relatedness between the generic concept (password) and the concepts of the domain ontology. If the threshold t=0.4, the algorithm will return the list [(1) web-customer password, (2) web-customer name].

Table 8.1. Generic/ domain concepts similarity.

Password	Web-customer password	Web-customer name	Wishing item
Sim (password, C_{domain})	0.75	0.41	0.26

Finally, if no concept of the domain corresponds semantically to the input generic concept, then the algorithm will return a list of all domain concepts (line 19). In other words, it tells

the user that no concept in the domain corresponds to what you are looking for, but this is the list of the domain concepts in case any of them interests you.

2 SECURITY REQUIREMENTS ELICITATION RULES

Given the security goal "Maintain the confidentiality of web-customer password", the engineer (Zach) wishes to know what are the potential threats? The vulnerabilities? The attackers? The attack methods? The security requirements to mitigate the threats? Who are the actors related to this goal? He also wants to produce a Secure Tropos model to have a better view and analysis of his assets, vulnerabilities and security requirements. To overcome these issues, we defined a set of mapping and production rules that build the bridge between the security ontology, the security goals, domain ontologies, and produce Secure Tropos (SecTro) models.

Mapping rules are described under the form Concept $_{source}$→ Concept $_{target}$. A given concept in the source is mapped to another concept in the target, where source and target are either the security ontology, the domain ontology, the security goal or a SecTro concept.

For example, the rule: "Agent $_{Domain\ Ontology}$ → Actor $_{SecTro}$" means that the concept *Agent* in the domain ontology is mapped to the concept *Actor* in the SecTro model.

The rule Security criterion $_{security\ goal}$ → Security Objective $_{SecTro,}$ means that the concept Security criterion in the security goal is mapped to the concept security objective in the SecTro model.

Production rules are described under the form <S→C>, where S is a situation and C a conclusion. <S →C> means that if the situation S is meant, then the conclusion C can be drawn. The situation holds on input security goals, an input security ontology, and an input domain ontology. The situation is defined using a first order logic predicate that relies on two kinds of functions:

- EquivalentDomainConcept (Concept $C_{generic}$, Type-Concept, Ontology DomainOnt): where Concept $C_{generic}$ is a generic concept, Type-Concept is its type, DomainOnt is the domain ontology at hand. The function will return the concept(s) in the domain ontology that has(ve) the closest semantics to the concept $C_{generic}$ in the security ontology or the security goal. (This function is has been detailed in Figure 8.1)

- OntologyLink (Type, X, Y): is true if in the input ontology there is a link from concept X to concept Y that has the type <Type>. For instance OntologyLink (IsAffectedBy, X, Y) is true if in the ontology, X and Y are related by an "affects" link from Y to X.

Conclusions indicate elements that should be added to the output Secure Tropos model. There are two conclusions-functions:

- CreateConcept (ClassC, X): indicates that a concept X that instantiates the <ClassC> class should be created in the model.
- CreateLink(LinkTypeL, X, Y): indicates that a link from X to Y, of type <LinkTypeL>, should be created in the model.

Using the inputs: the domain ontology (online shopping) presented in chapter 7, the core security ontology presented in chapter 5 and the security goal (maintain confidentiality of user's information) presented below, the rules will produce a (possible) security requirements specification model. The following gives a detailed application of the rules to the running example.

2.1. GOAL AND ASSETS ANALYZES

The security requirements analysis process starts by analyzing the context: the organization, its agents, its assets and different goals. In Secure Tropos, this is done thanks to the concepts *organization, actors, goals,* and *resource.*

A. Organization identification

The concept *organization* in the security ontology is mapped to the concept *organization* in Secure Tropos (SecTro) model, and the concept *organization* in the domain ontology. Applying the *EquivalentDomainConcept* function will return the organizations that are in the online shopping domain ontology (here: "online shopping company"). The function CreateConcept will create in the SecTro specification the concept "Online shopping company". Table 8.2 summarizes the mapping and production rules (columns 1 and 2). The last column illustrates one possible part of the resulting SecTro model. In this case, it illustrates the creation of the organization (online shopping company) concept in SecTro model.

Table 8.2. Organization identification

Mapping	Production	Model (a possible part)
Organization $_{\text{security ontology}}$ → Organization $_{\text{domain ontology}}$ Organization $_{\text{domain ontology}}$ → Organization $_{\text{SecTro}}$	EquivalentDomainConcept (**Input**: organization x, online shopping domain ontology) (**Output**: Online shopping company) CreateConcept (Organization, Online Shopping Company)	

B. Actors and goals identification

Once the organization is identified, the SecTro's organizational view is enriched. This view contains actors and goals dependencies between them.

The *asset* (user's information) identified in the security goal is mapped to the *object* (web-customer password) in the domain ontology. Thanks to the OntologyLink (*RelatedObject*) in the domain ontology, the action (Register web customer) is identified. This action is mapped to a goal in the SecTro model. The function CreateConcept will create the goal "Register web customer". An agent in the domain ontology is mapped to an actor in the SecTro model. Thanks to the OntologyLink (RequireAction) and (PerformAction), the *actors* (web-application) and (web-customer) are identified. The function CreateConcept will create the two actors (web-application) and (web-customer). The function CreateLink will create *dependency* relations between the identified actors and the identified *goal*. Table 8.3 summarizes the mapping and production rules (columns 1 and 2) for actor and goals identification. The last column illustrates one possible part of the resulting SecTro model. In this case, it illustrates the creation of the actors (web-customer, web-application) and the goal (register web customer).

Table 8.3. Actors and goal identification

Mapping	Production	Model (a possible part)
Asset security goal → Object domain ontology Object domain ontology → Resource SecTro Action domain ontology→ Gaol SecTro Agent domain ontology → Actor SecTro	EquivalentDomainConcept (**Input**: user's information, Asset, online shopping domain ontology) (**Output**: Web-customer password) OntologyLink (*RelatedObject*, Web-customer-password, Register web-customer) CreateConcept (Goal, Register web-customer). OntologyLink (*PerformAction*, Register web-customer, Web-application) and OntologyLink (*RequireAction*, Register web-customer, Web-Customer). CreateConcept (Actor, Web-application) CreateConcept (Actor, Web customer) CreateLink (*Dependency*, Web-application, Register web-customer) and CreateLink (*Dependency*, Web-application, Register web-customer)	

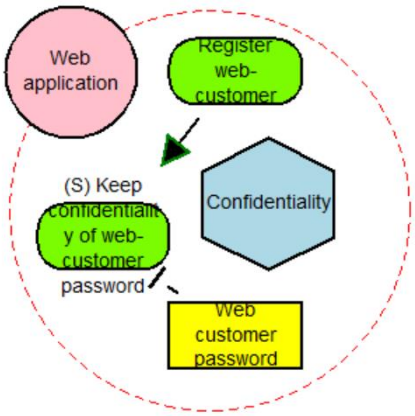
C. Security goals identification

The *security goal* (Keep confidentiality of web-customer password) is mapped to an *organizational goal* in SecTro model. According to Secure Tropos's concrete syntax, both organizational goals and security goals are expressed by the concept goal because both of them (security goals as well as organizational goals) can derive security requirements. Nevertheless, it is possible to identify a security goal from an organizational goal by adding an (S), and that what was adopted.

The security criterion expressed in the security goal (confidentiality) is mapped to a *security objective*. The function CreateConcept will create the concepts *goal* (Keep confidentiality of web-customer password), the *security objective* (confidentiality), and the *resource* (Web customer password). The function CreateLink will relate the *goal* (Keep confidentiality of web-customer password) to the organizational goal (Register web customer) using the relation MeansEnd.

Table 8.4 summarizes the mapping and production rules (columns 1 and 2) for actor and goals identification. The last column illustrates one possible part of the resulting SecTro model. In this case, it illustrates the creation of the security goal ((S) keep confidentiality of web-customer password), the organizational goal (Register web-customer), the security objective (Confientiality), the resource (Web customer password), as well as the necessary relations in the model.

Table 8.4. Security goal identification.

Mapping	Production	Model (a possible part)
[Verb][Security criterion][Asset] Security goal → Goal SecTro Asset Security goal → Ressource SecTro Security criterion Security goal → Security objective SecTro	OntologyLink (*PerformAction*, Register web-customer, Web-application) CreateConcept (Enhanced actor model, Enhanced Web-application model) CreateConcept (Goal, Register web-customer). CreateConcept (Goal, (S) Keep confidentiality of web-customer password). CreateLink (*MeansEnd*, Register web-customer, (S)Keep confidentiality of web-customer password) CreateConcept (Resource, Web-customer password). CreateLink (*Decomposition*, (S)Keep confidentiality of web-customer password, web-customer password); CreateConcept (Security objective, Confidentiality)	

2.2. THREAT ANALYSIS

The second step is to identify the potential threats, attackers, and attack methods, vulnerabilities. Here too, the security ontology and the domain ontology (using the rules) will guide the analysis and identification.

The concept *threat* in the security ontology is mapped to the concept *threat* in the SecTro model. The concept *threat agent* in the security ontology is mapped to the concept *malicious agent* in the SecTro model. Similarly, the concept *agent* in the domain ontology is mapped to the concept *malicious agent.* This is because we consider any agent in the domain as a potential attacker.

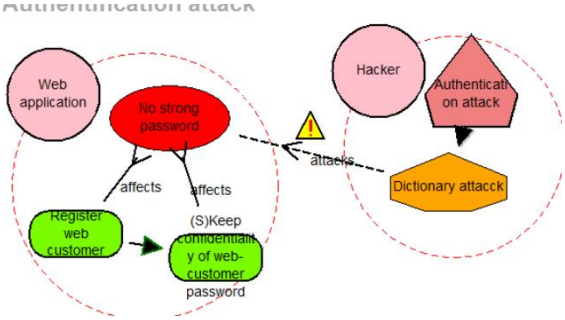
The concept *attack method* in the security ontology is mapped to the concept *attack method* in SecTro model. The concept *vulnerability* in the security ontology is mapped to the concept *vulnerability* in SecTro model.

The function OntologyLink identifies an "*Affect*" relation in the security ontology between the *threat* (authentication attack) and the *security criterion* (confidentiality). The function CreateConcept creates the *threat* (authentication attack) in the SecTro model. The function CreateLink creates the link *Impacts* that relates the threat (authentication attack) to the goal (Register web-customer).

With the same way, the concepts malicious actor (Hacker), attack method (Dictionary attack), and vulnerability (no strong password) are created and related with each other and with the rest of the concepts through the appropriate relations.

Table 8.5 summarizes the mapping and production rules (columns 1 and 2) for threat analysis. The last column illustrates one possible part of the resulting SecTro model. In this case, it illustrates the creation of the threat (authentification attack), the malicious actor (Hacker), the attack method (dictionary attack) as well as the necessary relations (attacks) and (affect) in the model.

Table 8.5. Threat analysis

Mapping	Production	Model (a possible part)
Security criterion security goal → Security criterion Security ontology Threat Security ontology → Threat SecTro Threat agent Security ontology → Malicious Actor SecTro Agent domain ontology → Malicious Actor SecTro Attack method security ontology → Attack method SecTro Vulnerability security ontology → Vulnerability SecTro	OntologyLink (*Affect*, authentication Attack, Confidentiality) CreateConcept (Threat, authentication attack) CreateLink (*Impacts*, authentication attack, Register web customer) CreateLink (*Impacts*, authentication attack, web-customer password) OntologyLink (*LedBy*, authentication attack, hacker). CreateConcept (Malicious Actor, hacker) OR CreateConcept (Malicious Actor, Agent) /*in case an agent from the domain ontology is chosen*/ OntologyLink (*UseMethod*, authentification attack, dictionary attack). CreateConcept (Attack method, dictionary attack) CreateLink (*Embedded*, authentication attack, dictionary attack) OntologyLink (*Exploits*, authentication attack, no strong password) CreateConcept (Vulnerabity , no strong password) CreateLink(*Attacks*, dictionary attack, no strong password) CreateLink (*Affects*, no strong password, Register web customer) CreateLink (*Affects*, no strong password, (S) Keep confidentiality of web-customer password)	

2.3. SECURITY REQUIREMENTS ANALYSIS

The final step is concerned about analyzing the appropriate security requirements that mitigate the identified threats in the previous step.

Applying the function OntologyLink on the relation *MitigatedBy* the security ontology will allow the identification of the two security requirements:

Req1. "(The agent)$_{agent}$ should (lock)$_{Action}$ (accounts)$_{Asset}$ (after reaching logon failure threshold) $_{under\ which\ condition}$".

Req2. (The agent)$_{agent}$ should (validate)$_{Action}$ (password)$_{Asset}$ (minimum length and complexity) $_{additional\ information}$".

The function EquivalentDomainConcept will help to make these two requirements more domain-specific by replacing the *asset* (accounts) by (web-customer accounts), and the *asset* (password) by (customer password). That is why the mapping rule maps the concept *asset* to *object* in the domain ontology. Therefore, the new security requirements are:

Req1. "(The agent) $_{agent}$ should (lock) $_{Action}$ (web customer accounts) $_{Asset}$ (after reaching logon failure threshold) $_{under\ which\ condition}$".

Req2. (The agent) $_{agent}$ should (validate) $_{Action}$ (web customer password) $_{Asset}$ (minimum length and complexity) $_{additional\ information}$".

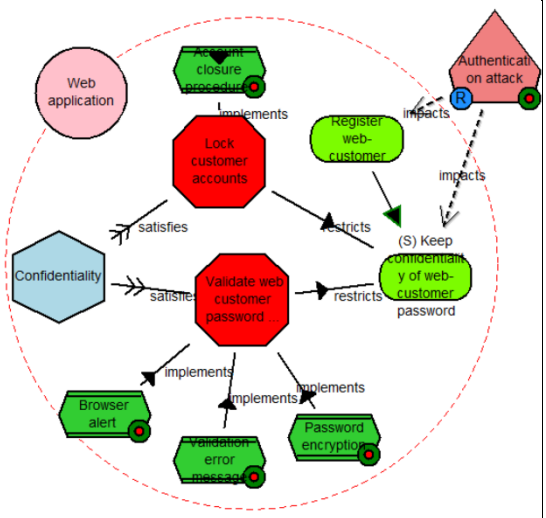
To model these two security requirements in the SecTro model, another mapping rule maps the part of the requirement ([Action][Asset] <Additional information>) to the concept *security constraint* in the SecTro model.

In this example, the function CreateClass will creates two *constraints*: "Lock web-customer accounts after reaching logon failure threshold" and "Validate web-customer password minimum length and complexity". The function CreateLink will relate these two created constraints to the security *goal* ((S) Keep confidentiality of web-customer password) through the relation *restricts*. It relates the *security constraint* (confidentiality) to the created security goals through a *satisfies* relation.

Applying the function OntologyLink on the relation *Fulfil*, the security ontology will allow identification of controls (security mechanisms in SecTro). Hence, the security mechanisms (browser alert, validation error message, and password encryption) are created and linked to the security constraint (validate password minimum length and complexity).

Table 8.6 summarizes the mapping and production rules (columns 1 and 2) for security requirements analysis. The last column illustrates one possible part of the resulting SecTro model.

Table 8.6. Security requirements analysis

Mapping	Production	Model (a possible part)
Security Requirements $_{\text{Security ontology}}$ = <When, under which conditions> [The agent][should/shall/ will] [Action][Asset $_{\text{Security ontology}}$] <Additional information>. Asset $_{\text{Security ontology}}$ → Object $_{\text{Domain ontology}}$ [Action][Object $_{\text{Domain ontology}}$ <Additional information>. → Constraint $_{\text{Sectro}}$ Control $_{\text{security ontology}}$ →Security mechanism $_{\text{SecTro}}$	OntlogyLink (*MitigatedBy*, authentication attack, The agent should lock accounts after reaching logon failure threshold) OntlogyLink (*MitigatedBy*, authentication attack, The agent should validate password minimum length and complexity) EquivalentDomainConcept (**Input**: accounts, Asset, online shopping domain ontology) (**Output**: web-customer accounts) EquivalentDomainConcept (**Input**: password, Asset, online shopping domain ontology) (**Output**: web-customer password) CreateClass (Security constraint, lock web-customer accounts after reaching logon failure threshold). CreateLink (*Restricts*, Lock web-customer accounts after reaching logon failure threshold,(S)Keep confidentiality of web-customer password) CreateClass (Security constraint, validate web customer password). CreateLink (*Restricts*, Validate web customer password, (S)Keep confidentiality of web-customer password) OntlogyLink (*Fulfils*, Password encryption, The agent should validate password minimum length and complexity) CreateClass (Security mechanism, Password encryption). CreateLink (*Implements*, Password encryption, validate web customer password). CreateClass (Security mechanism, validation error message). CreateLink (*Implements*, validation error message, validate web customer password). CreateClass (Security mechanism, browser alert). CreateLink (*Implements*, security mechanism, validate web customer password). CreateClass (Security mechanism, account closure procedure). CreateLink (*Implements*, account closure procedure, lock customer accounts). CreateLink (*Satisfies*, Confidentiality, validate web customer password). CreateLink (*Implements*, Confidentiality, lock customer accounts).	

The mapping rules and the production rules can be applied iteratively until all the possible artifacts chosen by the requirement engineer have been added into the requirements model specification. The requirements identification process can be considered as complete when no new security goal can be added in the specification. The security requirements specification model can be considered complete when no new security threat or asset can be added.

3 PRODUCING TEXTUAL SECURITY REQUIREMENTS (SPECIFICATION DOCUMENT)

Once his model has been constructed and his analysis finished, Zach would like to derive textual security requirements and organize them into a specification document.

We propose to organize the information captured during the whole analysis process as follows. For each security criterion, the related identified assets are elicited, followed by the identified security requirements enumerated one by one.

For each security requirements, the two rules are applied:

- Actor $_{\text{Sectro}}\rightarrow$ Agent $_{\text{security requirement}}$

By default, the *actor* for whom the identified constaints are substituted will replace the *agent* in each security requirement. In our example, the actor (Web application) will be the agent of the security requirements (Req1 and Req2).

- Agent $_{\text{Domain ontology}}\rightarrow$ Agent $_{\text{security requirement}}$

Moreover, it is also possible to delegate the requirement to any other agent in the domain ontology (for e.g. web developer).

Confidentiality:

Assets: Web customer password.

Req1. The web application should lock web-customer accounts after reaching logon failure threshold.

Req2. The web application should validate web-customer password minimum length and complexity.

Figure 8.2. Part of the security requirements specification document.

4 CONCLUSION

This chapter has presented an important part of AMAN-DA. It has addressed the question of how to put together an input security ontology, a domain ontology and given security goals. The algorithm and the different rules presented make it possible to perform the ontology based domain security requirements engineering process. The implementability as well as the efficiency and ease of use of the proposed rules remains open for the moment. The chapters to follow will evaluate AMAN-DA. Appendix D proposes an implementation into a tool.

Part 3

Evaluation

Chapter 9

Case study: Security requirements for the maritime domain (and other domains)

Case studies are a good empirical technique to evaluate methods in real situations. By definition, a case study "*investigates a contemporary phenomenon within its real-life context*" (Lethbridge, 2001). (Robson, 2011) consider case studies as "*a strategy for doing research that involves an empirical investigation of a particular contemporary phenomenon within its context using multiple sources of evidence*."

The first part of this chapter reports the application of AMAN-DA in a real case study that was undertaken to address the question "*Is it possible to elicit domain specific security requirements using AMAN-DA?*" The case study demonstrates the capability of AMAN-DA to handle security requirements engineering in a specific domain of application – here, the maritime domain. The second part of the chapter addresses the question: *Can AMAN-DA be applied to any domain?* It examines the application of AMAN-DA with two other domains and generalizes the results to different domains.

1 CASE STUDY RESEARCH PROCESS AND GUIDELINES

The research process proposed by (Runeson et al., 2012) was used and adapted for conducting the case study. It is based on five major steps as described in Figure 9.1.

1. ***Case study design*** – Objectives are defined and the case study is planned.
2. ***Preparation for the data collection*** – procedures and protocols for the data collection are defined.
3. ***Collecting evidence*** – data collection procedures are executed on the case studied.
4. ***Analysis of collected data*** – data analysis procedures are applied to the data.
5. ***Reporting*** – the study and its conclusions are packaged in feasible formats for reporting.

Figure 9.1. Case study process adapted from (Runeson et al. 2012)

The case study process is a flexible design strategy; there is a significant amount of iteration over the steps. The data collection and analysis may be conducted incrementally. If insufficient data are collected for the analysis, more data collection may be planned. However, there is a limit to the flexibility; the case study should have specific objectives set out from the beginning. If the objectives change, it is a new case study rather than the existing one. During the collecting evidence step, data collection procedures are applied to the data. Once the data is analyzed, the study and its conclusions are packaged in a feasible format for reporting.

In addition to this process, (Runeson et al., 2012) report that researchers have recognized a range of elements that need to be considered in the design of a case study (Figure 2)

1. Rationale. Why is the study being done?
2. Purpose. What is expected to be achieved with the study?
3. The case. Overall, what is being studied?
4. Units of analysis. In more detail, what is being studied?
5. Theory. What is the theoretical frame of reference?
6. Research questions. What knowledge will be sought or expected to be discovered?
7. Propositions. What particular (causal) relationships are to be investigated?
8. Define concepts and measures. How are entities and attributes being defined and measured?
9. Methods of data collection. How will data be collected?
10. Methods of data analysis. How will data be analyzed?
11. Case selection strategy. How will cases (and units of analysis) be identified and selected?
12. Data selection strategy. How will data be identified and selected. For example, who will be interviewed?
13. Replication strategy. Is the study intended to literally replicate a previous study, or theoretically replicate a previous study, or is there no intention to replicate?
14. Quality assurance, validity and reliability. How will the data collected be checked for quality? How will the analysis be checked for quality?

Figure 9.2. Elements of the research design adapted from (Runeson et al. 2012)

These elements in addition to the research process were considered during the elaboration of the maritime case study as detailed in the next sections.

2 MARITIME CASE STUDY

2.1. CASE STUDY DESIGN

A. Rationale: *Why was the study done?*

The study was undertaken at the stage of the research project, after the AMAN-DA method was proposed. The research group (PhD student and supervisors) was interested to see how AMAN-DA would apply in a real context and with a particular domain of application.

B. Purpose: *What is expected to be achieved with the study?*

After having applied AMAN-DA on small-scale examples (e.g. the online shopping running example reported earlier), the team was interested in using AMAN-DA for a large-scale case study. A domain expert was interviewed and then involved in the validation of the results obtained from AMAN-DA. We decided to focus on the use of AMAN-DA to capture stakeholder's security goals in a maritime context and elicit security requirements specific to the maritime domain as part of the modeling of the information system of a bulk carrier[13] independently of the other information systems of maritime transport.

C. Case and Units of Analysis: *What is being studied?*

The study is part of the elaboration of the ship's information system, and focuses on the (security) requirements elicitation phase. The considered unit of analysis is a ship (a bulk carrier) that belongs to a maritime company, which manages other ships too. The ship's information system may interact with other systems such as the port facility information system or the company information system. Figure 9.3 presents the distinction between the context, the case and the unit of analysis within our case as recommended by Yin (Yin, 2014).

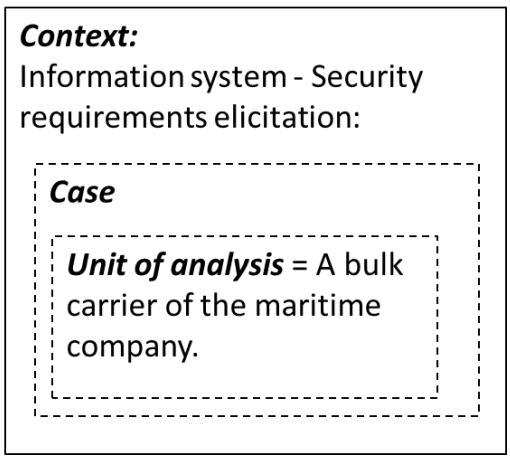


Figure 9.3. Overview of the context, case and unit of analysis according to (Yin 2014)

As main information[14], a ship is composed of two departments (Figure 9.4.):

- The deck department.
- The ship's engine department.

[13] A ship that carries non-liquid cargoes such as grain or ore in bulk, according to Oxford Dictionary.
[14] As the reader may notice, during the reported case study (for confidentiality and ethical considerations), the name of the company, the name of the captain, the various agents and any sensitive information are omitted for confidentiality's sake.

The deck department is managed by the captain, the second officer and their subordinates. Its main functions involve administrative tasks such as scheduling work, quality control, coordinating with other departments, and conflict resolution.

The ship's engine department is managed by the chief engineer, the second engineer and their subordinates. The engine department is responsible for all maintenance and operation of the electrical and mechanical equipment. Besides the engines in the engine room, the engine department crew is responsible for all of the sewage, air conditioning, lighting, and water on the ship[15]. In addition to those main departments, crew's cabins, offices, galley, and some stores are handled by the steward, the cook, the baker, the waiter and their subordinates.

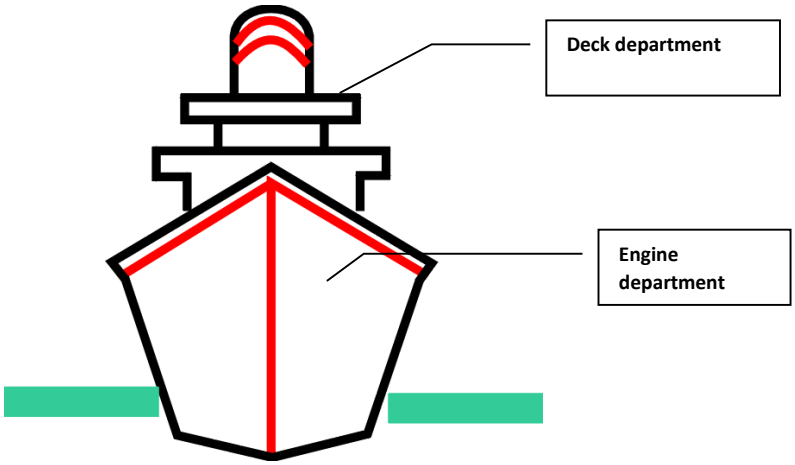


Figure 9.4. Ship departmental composition

D. Theoretical Frame of Reference: *What is the theoretical frame of reference?*

AMAN-DA is the theoretical framework of reference for this study. AMAN-DA looks into eliciting security requirements for specific domains. It relies on an input security ontology and in this case an input maritime domain ontology in addition to stakeholder's security goals.

- *Stakeholder's security goals*

AMAN-DA suggests formalizing stakeholder's security goals according to the security goal template model depicted in Chapter 6.

- *Maritime domain ontology*

The maritime domain ontology is the second input of AMAN-DA. The ontology has been constructed from many maritime sources in addition to interviews with experts from the maritime domain. It contains 85 concepts and 115 relationships. This ontology was shown in (Chapter 7- Figure 7.4).

[15] http://www.wisegeek.com/what-is-the-engine-department-on-a-us-merchant-ship.htm

- *Core security ontology*

The core security ontology presented in details chapter 5 was the third input of AMAN-DA within this case study. It has the necessary security-related knowledge for the security requirements process.

E. Research Questions and hypothesis*: What knowledge will be sought or expected to be discovered?*

As the goal of the case study is to evaluate the *feasibility* of AMAN-DA. The main research question we investigated during this case study was:

"***Is it possible to elicit maritime specific domain security requirements using AMAN-DA***?"

The null hypothesis (H0) that was tested is:

"H0: AMAN-DA is not capable of producing maritime domain specific security requirements"

2.2. PLANNING

A. Methods of Data Collection

The main source of information for this case study were gathered after long hours interviews with the captain. The interviews took 3 consecutive days (~ 15 hours) the first time to capture as much information as possible. Then there was a meeting by Skype once every two weeks for two months to review some points and clarify some ambiguities.

The interviews were performed as semi-structured interviews, more in the form of a discussion, using the interview instrument (a set of pre-prepared questions) as a guide of areas available to discuss. Some of the interviews were direct, while others were performed via Skype when the captain and our team were unable to meet up.

The interview instrument was created in advance and adapted slightly as the interviews progressed. Adaptations were primarily made with the purpose of gaining further information about security issues as they are seen on the ship. The interview instrument is presented in Appendix C.

The interviews concentrated mainly on understanding the structure of a ship, its departments, its employees (crews), and the different interactions between those parts, while seeking to identify some useful documents recommended by the captain that deal with security in the

maritime domain, as well as the main security goals of a ship as an organization during its travels as well as during its docking.

The captain suggested a couple of documents that were also consulted: (IMO, 1974) (IMO 2011) (Chebli, 2009).

Notes were taken during the interview, and sound recorded for later transcription. The interviews were thus stored and analyzed later.

B. Case and Data selection strategy

We had from the beginning, a strategic goal to look for a domain where security is a *critical* issue. It seems like the same cases are repeated over and over again in the literature. Most of those dealing with security were often related to the banking sector (Salinesi, Ivankina, and Angole, 2008) or health insurance sector (Mouratidis and Giorgini, 2007). The choice of a different case was intended to increase evidence, strengthen findings, and avoid biasing.

2.3. DATA ANALYSIS

Figure 9.5 presents an overview of the analysis process. After the interviews with the ship captain were recorded and notes were taken, the data that had been gathered was transcribed and categorized. We then applied AMAN-DA on the identified security goals. The results of the application (the different models and specifications generated) were discussed with experts. The next sections discuss in detail these steps:

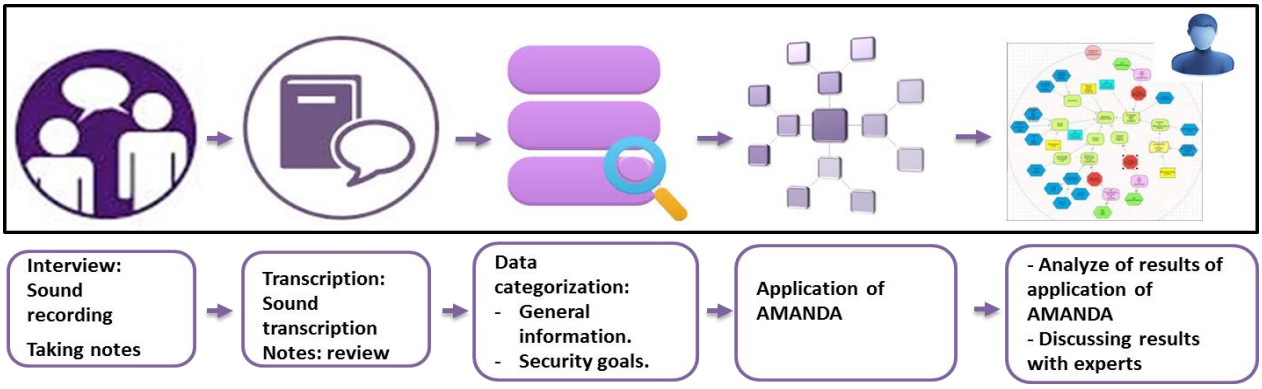


Figure 9.5. Overview of the analysis process.

Recording, transcribing and data categorization

Each hour of the interview took about two hours to be transcribed. The information gathered from the interview was mixed and concerns different aspects of the maritime organization on the ship, as well as relating to different levels of abstraction. Despite being semi-structured

and with pre-prepared questions, the interview with the ship captain tended from time to time to go out of the scope of the study. In order to keep focus, the transcribed data was categorized into three categories:

- Information not useful to the case study;
- General information useful to the case study;
- Security goals.

Some information was found to be useless to the case study and to the requirements elicitation phase. Other data were general information useful for our own understanding of the ship organization only. The last category was clear security goals expressed by the ship's captain. Only these were formalized using the AMAN-DA security goal linguistic template and were part of the input of the method. AMAN-DA was then applied on the security goals captured. The outputs of the application were the different potential threats, vulnerabilities, security requirements that are likely to be present in the domain. The different SecTro models were generated as well as the final specifications.

Table 9.1 presents part of the data categorization step. The transcribed data was stored in a table with the three categories identified to ensure full traceability.

Table 9.1. Example table part corresponding to the interview with the ship's captain.

Category	Some parts of the interview
Information not useful to the case study	"I worked for passenger ships then moved to bulk carriers, it is much easier to work with products than people!"
General useful information	"A ship is composed of the deck department and the engine department. The deck department is managed by the captain, the second officer and their subordinates. The ship's engine department is managed by the chief engineer, the second engineer and their subordinates."
Security goals	"When communicating with other ships, the captain need to be sure of the identity of his interlocutors, especially in areas of sensitive navigation" "During navigation, the captain would like to maintain the meteorological data, the nautical charts, the ocean current maps, and the logbooks available"

The next two sections present the application of AMAN-DA on some captured security goals.

A. Application of AMAN-DA

A.1. Inputs of AMAN-DA for the case study

7 security goals (SG_i) were captured during the initial interview. These goals are formalized according to the security goal template model of AMAN-DA.

1. (SG_1) The chief engineer is asked to maintain the **confidentiality** and **integrity** of the ship's engine plans.
2. (SG_2) The captain would like to maintain the **integrity** and **confidentiality** of the documents that he manages (ship's certificates, ship's plans, and communication certificates). These documents can be consulted by the SSO (Ship Security Officer).
3. (SG_3) Due to previous incidents, the captain would like to maintain the availability of the communication equipments on board such as the VHF radio, the talkie walkies.
4. During navigation, (SG_4) the captain would like to maintain the availability of the meteorological data, the nautical charts, the ocean current maps, and the logbooks.
5. (SG_4) The company would like to keep the evaluations of the ship's crews **confidential**; these evaluations allow for recruitment of a crew for other missions.
6. (SG_5) During loading/unloading of goods, the captain would like to preserve the **integrity** of the bill of lading document, the stability booklet and the stowage plan. (SG_6) He would also like to make them **available** for the department members.
7. (SG_7) The system should ensure the **non-repudiation** of communication between the ship and the other ships navigating in the same area.

A2. Outputs of AMAN-DA for the case study

Through the use of AMAN-DA, the security requirements (Req_i) that materialize the input security goals (SG_i) are defined. This is done after analyzing the valuable ship assets to protect, and the potential risk analysis (threats (T_i), vulnerabilities (V_i)). The corresponding Secure Tropos models are generated as well as the final specification.

- Analysis

Tables (9.2 to 9.5) present a potential analysis scenario of security goals (SG_{1-4}). With the four security goals as an input, the analysis through the use of AMAN-DA discovers 8 potential threats, 17 potential vulnerabilities, and 28 potential security requirements. Same analysis can be applied to SG_5 and SG_6.

Table 9.2. Analysis for SG1

Organization	Maritime organization.
Security Goal (SG_1)	(Maintain) $_{\text{verb}}$ the (confidentiality) $_{\text{Security criterion}}$ of the (engine plans) $_{\text{asset}}$
Organizational goal	Maintain ship engine
Actors	Captain Chief engineer

Potential threats (selected)	(engine plans)
	T1. Unauthorized physical access.
Potential vulnerabilities (selected)	**T1.**
	V1. No key management. **V2.** No surveillance cameras. **V3.** No alarm system. **V4.** Open windows.

Potential security requirements (selected)	**V1.**
	Req1. The chief engineer should set proper termination procedures. **Req2.** The chief engineer should sign and account engine room keys, bridge keys, captain cabin keys and crew keys.
	V2.
	Req 3. The chief engineer should provide an adequate maritime surveillance system.
	V3.
	Req4. The chief engineer should equip the structures with ship security alarm system, machinery space alarm.
	V4.
	Req5. After working hours, the chief engineer should lock engine room windows.

Table 9.3. Analysis for SG2

Organization	Maritime organization.
Security Goal (SG_2)	(Maintain) $_{\text{verb}}$ the (integrity) $_{\text{Security criterion}}$ of the (ship certificates) $_{\text{asset}}$
Organizational goal	Manage ship's documents
Actors	Company director Captain

Potential threats (selected)	(ship certificates)	
	T1. Fire. **T2.** Uncontrolled flow of water	
Potential vulnerabilities (selected)	**T1.**	**T2.**
	V1. No testing of fire extinguisher **V2.** Open fire doors. **V3.** No fire suppression	**V4.** No regular testing of pipes **V5.** No water detector

Potential security requirements (selected)	**V1.**
	Req1. The ship security officer should test FlexiFog fire extinguishers, FlexiFoam fire extinguishers, and gaseous based fire extinguishers.
	V2.
	Req 2. After working hours, the ship security officer should lock bridge, engine and galley rooms' doors.
	V3.
	Req3. The ship security officer should provide an automated and tested maritime fire suppression system. **Req4.** The ship security officer should make available the fire extinguishers (FlexiFog, FlexiFoam, and gaseous fire extinguishers).
	V4.
	Req5. The ship security officer should test the ship pipes.
	V5.
	Req6. The ship security officer should provide an automated and tested water flood sensor to the relevant areas.

Table 9.4. Analysis for SG3

Organization	Maritime organization.
Security Goal (SG_3)	(Maintain) $_{verb}$ the (availability) $_{Security\ criterion}$ of the (communication equipments: VHF radio, talkie walkies) $_{asset}$
Organizational goal	Watch keeping the ship
Actors	Captain Duty officer

Potential threats (selected)	(Communication equipments)
	T1. Theft
Potential vulnerabilities (selected)	**T1.**
	V1. No use of locks **V2.** No regularly reviewed resource inventory.

Potential security requirements (selected)	**V1.**
	Req1. The duty officer should use adequately fixed marine padlocks.
	V2.
	Req 2. Regularly, the duty officer should make and update the bridge inventory, bridge documentation inventory, the navigation equipment inventory, ship radio station equipment inventory, replacement part inventory, paint inventory, cleaning equipment inventory.

Table 9.5. Analysis of SG4

Organization	Maritime organization.
Security Goal (SG_4)	(Maintain) $_{verb}$ the (availability) $_{Security\ criterion}$ of the (meteorological data, ocean current maps) $_{asset}$
Organizational goal	Manage weather data and equipment.
Actors	Captain Weather officer

Potential threats (selected)	(meteorological data)			
	T1. Configuration error. **T2.** Untrained personnel. **T3.** Failure of systems. **T4.** Network attack.			
Potential vulnerabilities (selected)	**T1.**	**T2.**	**T3.**	**T4.**
	V1. No change of preset password. **V2.** Insecure installation and configuration of password.	**V3**. Lack of training.	**V4.** No backups. **V5.** Insufficient maintenance.	**V6.** No secure internet connexion.

Potential security requirements	**V1.**
	Req1. The weather officer should control the allocation of officers' passwords

(selected)	through a formal management process.
	V2.
	Req2. The weather officer should ensure appropriate installation and configuration of ship manager software.
	V3.
	Req3. The captain should ensure good training of the ship officers. **Req4.** The captain should define clear roles and responsibilities of ship officers.
	V4.
	Req5. The weather officer should perform backups of old configurations. **Req6.** The weather officer should provide appropriate application data backup procedures.
	V5.
	Req7. The weather officer should ensure the maintenance of applications. **Req8.** The weather officer should ensure the existence of maintenance contracts for bridge applications.
	V6.
	Req9. The weather officer should update and review maritime specific mechanisms to secure communication.

- Secure Tropos model generated

The following figures represent the Secure Tropos models generated after applying AMAN-DA to the previous security goals ($SG_{1\text{-}4}$). Figures 9.6 to 9.11 represent the Secure Tropos models generated by the use of AMAN-DA. These models correspond to the analysis made in the previous tables.

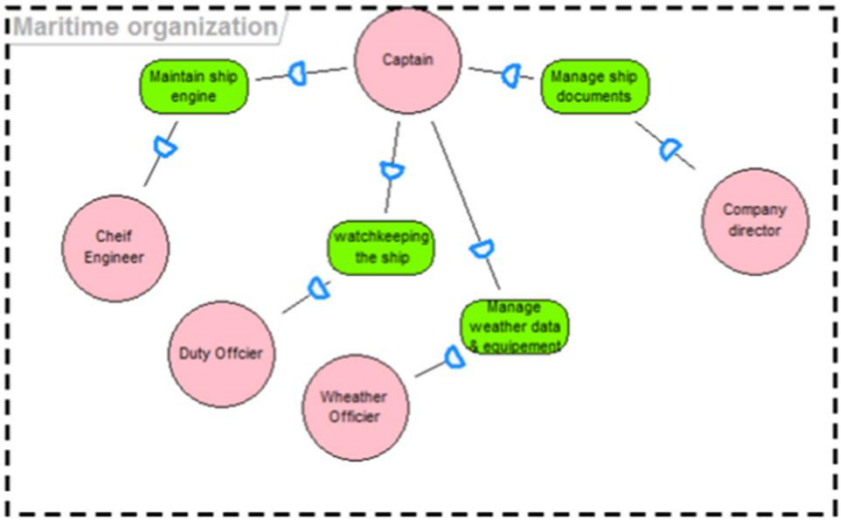


Figure 9.6. Part of the organizational view

Figure 9.6 represents part of the organizational view generated. It contains the main actors and the goal dependencies between them as well as the name of the organization.

Figure 9.7 represents part of the security requirements view that corresponds to the security goal (SG_1). This view includes the main security requirements to consider (security constraints in Secure Tropos).

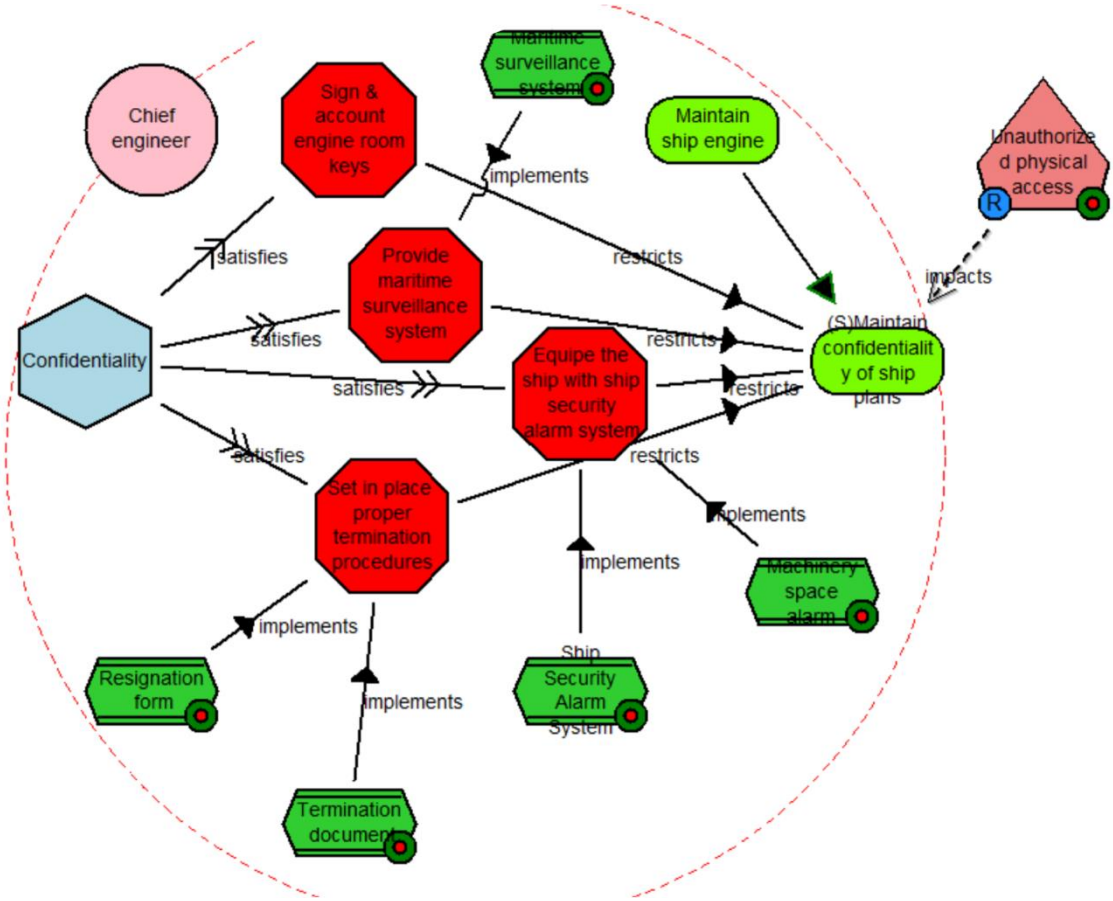


Figure 9.7. Part of the Security requirements view (SG1)

Figure 9.8 represents the Secure Tropos attack view; it displays the threat (unauthorized physical access), the attack method (social engineering) and vulnerabilities exploited (V_1 to V_4).

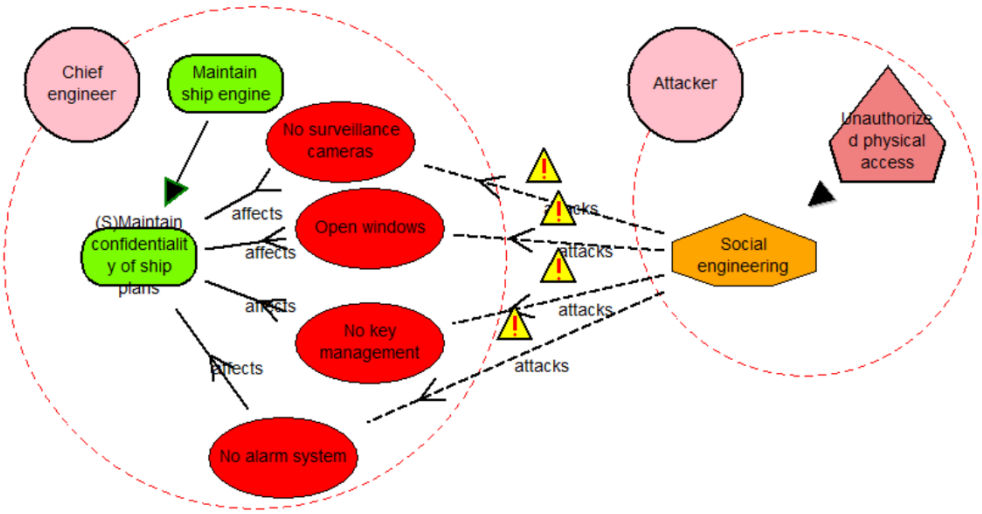


Figure 9.8. Part of the attack view (SG1)

Figures 9.9, 9.10, and 9.11 represent the security requirements views generated according to security goals ($SG_{2,\ 3\ and\ 4}$). The generated XML file that represents these models can be consulted in Appendix E.

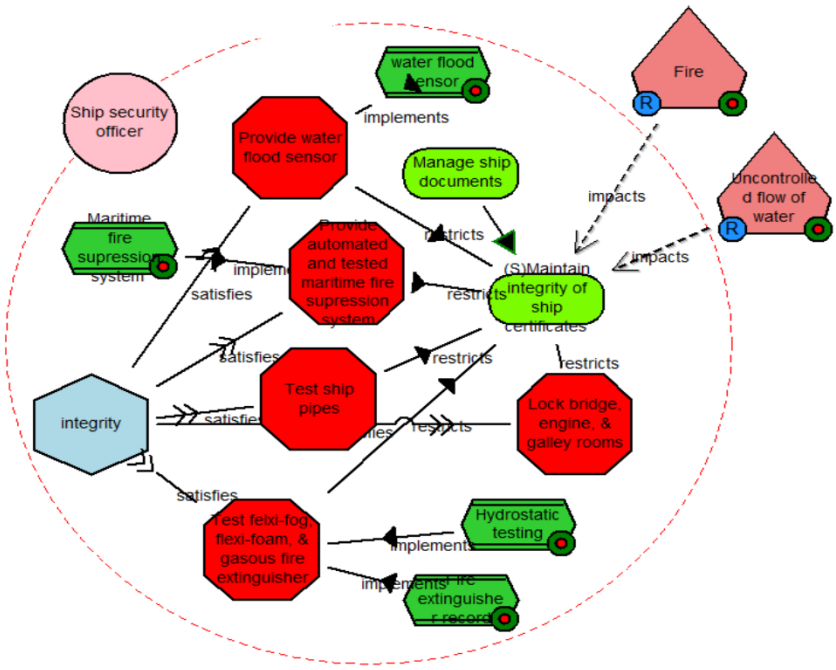


Figure 9.9. Part of the security requirements view (SG2)

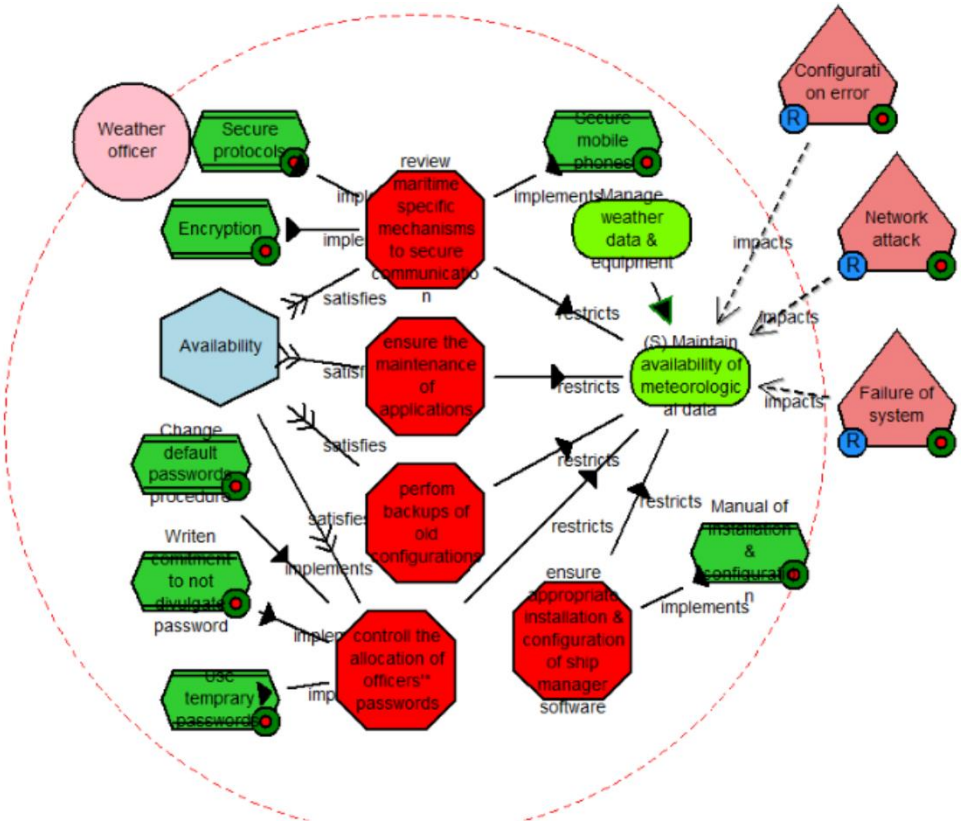


Figure 9.10. Part of the security requirements view (SG3)

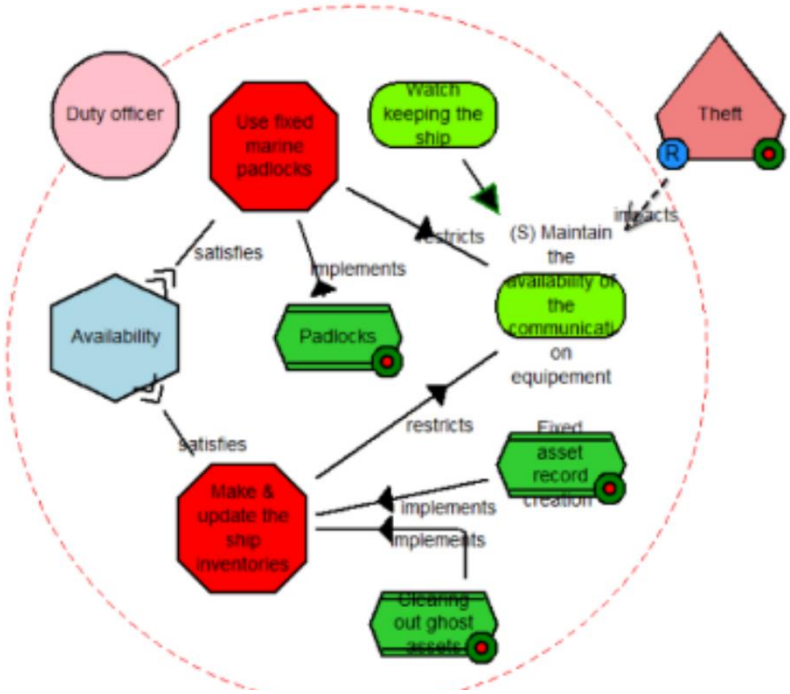


Figure 9.11. Part of security requirements view (SG4)

- Generated Specification

The other output of AMAN-DA is the textual specification. Figure 9.12 represents part of the specification generated; it contains the name of the organization, the different assets to protect, the potential threats, the potential vulnerabilities and the security requirements to consider.

```
Organization:    Maritime organization
_____________________________________________________________________

Asset: engine plans

Security Criterion: Confidentiality

Threat: Unauthorized physical access.

Vulnerabilities:

-> No key management
      Security Requirements:
      (Req1.) The chief engineer should set proper termination
procedures.
     (Req2.) The chief engineer should sign and account engine
room keys, bridge keys, captain cabin keys and crew keys.

-> No surveillance cameras
      Security Requirements:
     (Req3.) The chief engineer should provide an adequate
maritime surveillance system.

-> No alarm system
      Security Requirements:
     (Req4.) The Chief engineer should equip the structures with
ship security alarm system, machinery space alarm.

**********************************************************************

Asset: ship certificates

Security Criterion: integrity

Threat: Fire

Vulnerabilities:

-> No testing of fire extinguisher
   Security Requirements:
   (Req1.) The ship security officer should test FlexiFog fire
extinguishers, FlexiFoam fire extinguishers, and gaseous based
fire extinguishers.

-> Open fire doors
   Security Requirements:
  (Req2.)After working hours, the ship security officer should
lock bridge, engine and galley rooms' doors.

->No fire suppression
```

```
   Security Requirements:
   (Req3.) The ship security officer should provide an automated
and tested maritime fire suppression system.
  (Req4.) The ship security officer should make available the fire
extinguishers (FlexiFog, FlexiFoam, and gaseous fire
extinguishers)
-> No regular testing of pipes
   Security Requirements:
  (Req5.) The ship security officer should test the ship pipes.

-> No water detector
   Security Requirements:
   (Req6.) The ship security officer should provide an automated
and tested water flood sensor to the relevant areas.
```

Figure 9.12. Part of the textual specification

2.4. DISCUSSING RESULTS WITH EXPERTS

The data analysis process ended up with discussions on the results of AMAN-DA's application (the generated security requirements models and specifications) with a group of four experts. The group comprises: the domain expert (the ship's captain), the requirements engineering expert (a scientist known in the requirements engineering community), and two security standardization experts who served as security experts. The discussion was carried out face to face with the experts during one meeting. First, the context of the project and the method were presented to them. Second, experts reviewed the SecTro model and security requirements produced by the application of AMAN-DA.

The results of this meeting can be summarized as follows:

a. The main recurrent comments made by the experts were that the method and its idea of automation are "very" (the adverb was employed several times) "interesting", and "important".
b. From the results of application of AMAN-DA in the maritime domain, experts consider that the method is capable of producing domain specific security requirements but asked to see more applications to other domains for a better generalization of this statement.
c. One expert was interested in the richness of the security requirements modeling language chosen (Secure Tropos in our case) and whether it was enough to model all the domain concepts. Also, why is it the only language that the method handles?
d. Experts (security and requirements) commented on the positive results of AMAN-DA compared to a case study undertaken by RITA in the banking sector where the

resulting requirements were too generic and not domain specific. The experts recognized the capacity of AMAN-DA to achieve a better capture of stakeholders' security goals, threat analysis, model and textual specifications generation.

e. Experts suggested evaluating the method with a controlled experiment with end-users to evaluate the ease of use and the usefulness of the method.

3 THREATS TO VALIDITY

The case study might be criticized for many aspects. The main threats to validity can be as follows:

- Although the choice of the captain as the person to interview was strategic, it turned out that interviewing only one person was not enough. In many cases, the captain mentioned that other persons who might be more knowledgeable about some other parts of the ship (e.g. the chief engineer who is an expert in the ship engine should be asked for example.)
- Security goals provided by the stakeholder were sometimes ambiguous. While we were concerned by the informational and organizational security goals, he mostly focused on the security of the life of individuals on board. The links between both were not always explicit.
- Discussing the results of the application of AMAN-DA with some experts was good but the analysis remains qualitative, which is not enough to evaluate the ease of use and the usefulness of AMAN-DA.
- The experts involved in discussing and analyzing the results comprise the supervisors, which may have left room for subjectivity.
- Due to the timing of the research, we could not replicate the case study in another domain. Application to only one domain and with one company might not be enough, other applications are necessary to generalize the results.

4 LESSONS LEARNED

The application of AMAN-DA to the case study might change established thought, in the sense that security knowledge by itself is not enough if it is not completed with domain knowledge. This result is a step forward to the research community. Vice versa, domain expertise alone remains weak if it is not reinforced by security knowledge. Recall that before

AMAN-DA, some propositions were limited to relying on generic security ontologies only RITA (Salinesi, Ivankina, and Angole, 2008), (Velasco et al., 2009) and their shortcoming was that the resulting security requirements were too generic and not specific to the domain at hand. Now the new open question is whether using both domain and security ontologies is sufficient or using domain specific security ontologies (i.e. constructing security ontologies for each domain and use them in AMAN-DA) would be a better solution?

The other point learnt or noticed is that, even with the security goal model proposed by AMAN-DA, the step of capturing security goals from stakeholders remains difficult: careful recording and transcribing of stakeholder interviews was needed before applying the method. This step could be improved in further research on how to conduct successfully an interview with stakeholders in order to capture security goals. For the researchers, another open question would be, what are the necessary parameters to take in consideration in the formulation of the security goals and security requirements? For instance, the level of security (i.e. low, medium, high) is an example to explore.

The case study has shown that AMAN-DA provides a methodological and organized way to produce specifications (c.f. Figures 4-7). The question here is what practitioners think about that, and whether AMAN-DA improves their practices. That is why a controlled experiment with them is the next natural step needed to be implemented.

5 APPLICATION TO OTHER DOMAINS

In this section we attempt to validate the property of genericity of the method. That is to say that AMAN-DA is a generic method and can be used for more than one domain, and more than one domain ontology. In fact, this section addresses the question:

RQ. *Can AMAN-DA be applied to any domain?*

Chapter 8 in this thesis has already presented an application of AMAN-DA with an online shopping domain ontology. The previous sections of the current chapter presented the application of AMAN-DA with a maritime domain ontology.

Now, let us examine two other applications: with a web publishing domain ontology, and a sales domain ontology.

6.1. APPLICATION OF AMAN-DA WITH A WEB PUBLISHING DOMAIN ONTOLOGY

- *Inputs of AMAN-DA*

Figure 9.13 presents the input ontology, a web publishing domain ontology that encapsulates some knowledge related to this domain. One can see the organisation (publishing company) which has a number of agents (editor, reader, reviewer, web publishing system). There are a number of objects in the ontology (web server, review, submitted article, edited article). Various actions are required and performed by the different agents (manage articles, submit review, search article, etc.)

The security goals are captured from stakeholders and formalized according to AMAN-DA:

SG1. Keep the integrity of the submitted articles.

SG2. Maintain the availability of the web server.

As the reader may recall, the third input of AMAN-DA is the core security ontology presented in chapter 5.

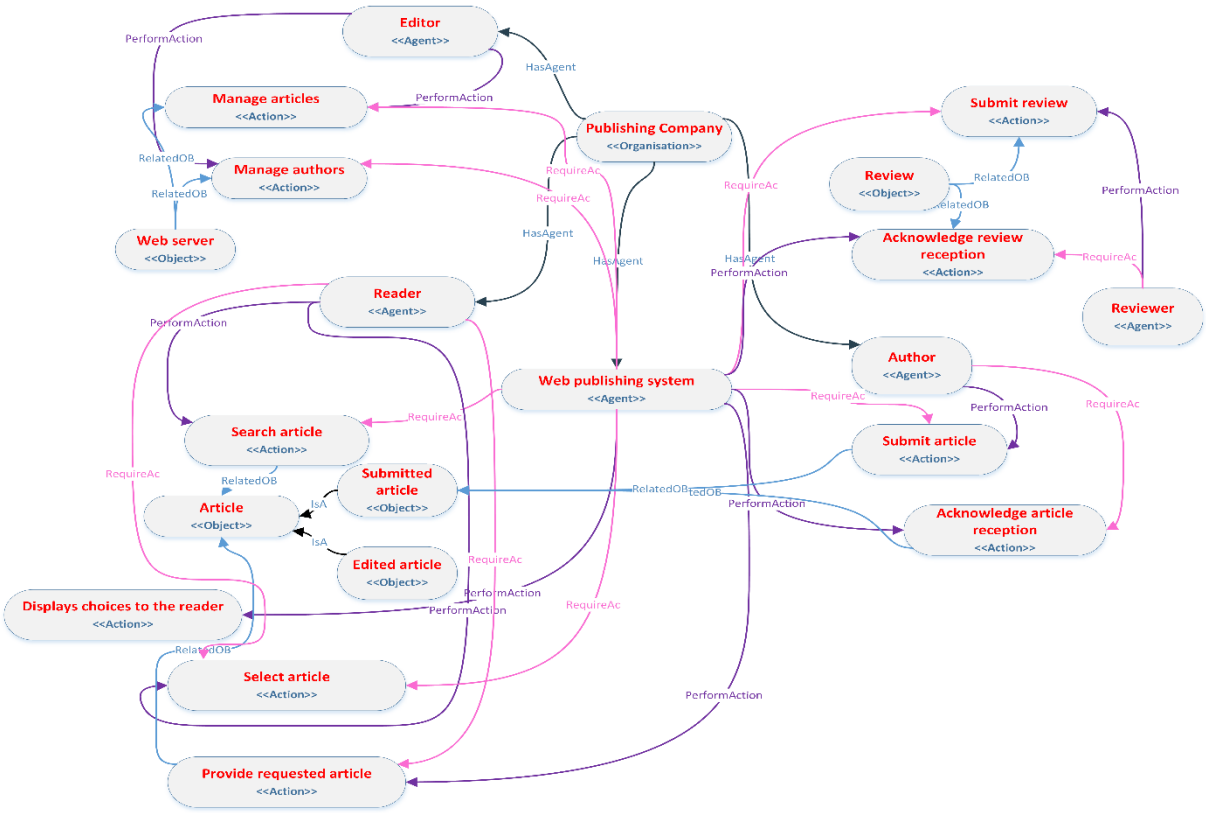


Figure 9.13. A web publishing domain ontology

- *Outputs of AMAN-DA*

Through the use of AMAN-DA, the security requirements (Req_i) that materialize the input security goals (SG_1 and SG_2) are defined. The following corresponding Secure Tropos models are generated as well as the final specification.

- Secure Tropos models generated

Figures 9.14 and 9.15 represent the Secure Tropos models generated after applying AMAN-DA to the previous security goals (SG_1 and SG_2).

Figure 9.14 presents part of the organizational view generated. It contains the main actors (Editor, Author, Reader, Web publishing system) and the goal dependencies between them as well as the name of the organization (Publishing Company).

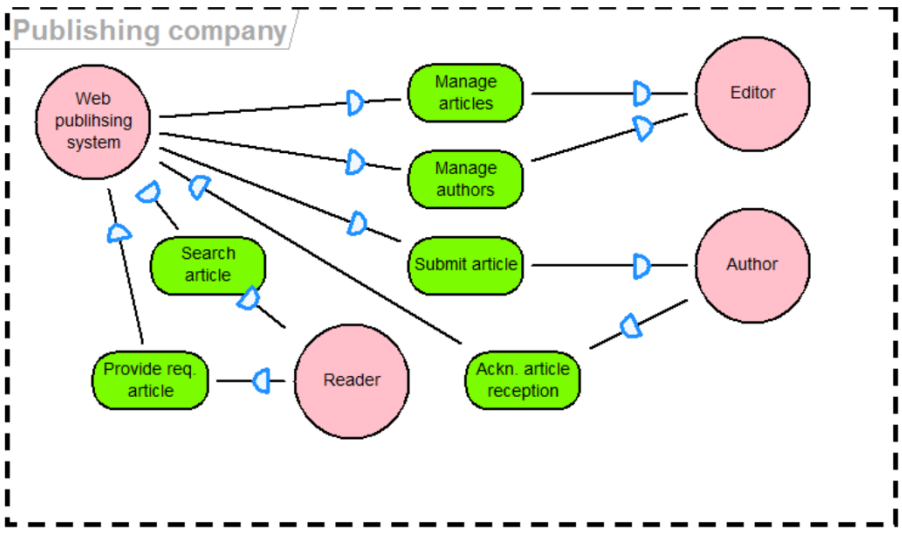


Figure 9.14. Organizational view - publishing domain

Figure 9.15 represents part of the security requirements view that corresponds to the security goal (SG_2). This view includes the main security requirements to consider (security constarints in Secure Tropos)

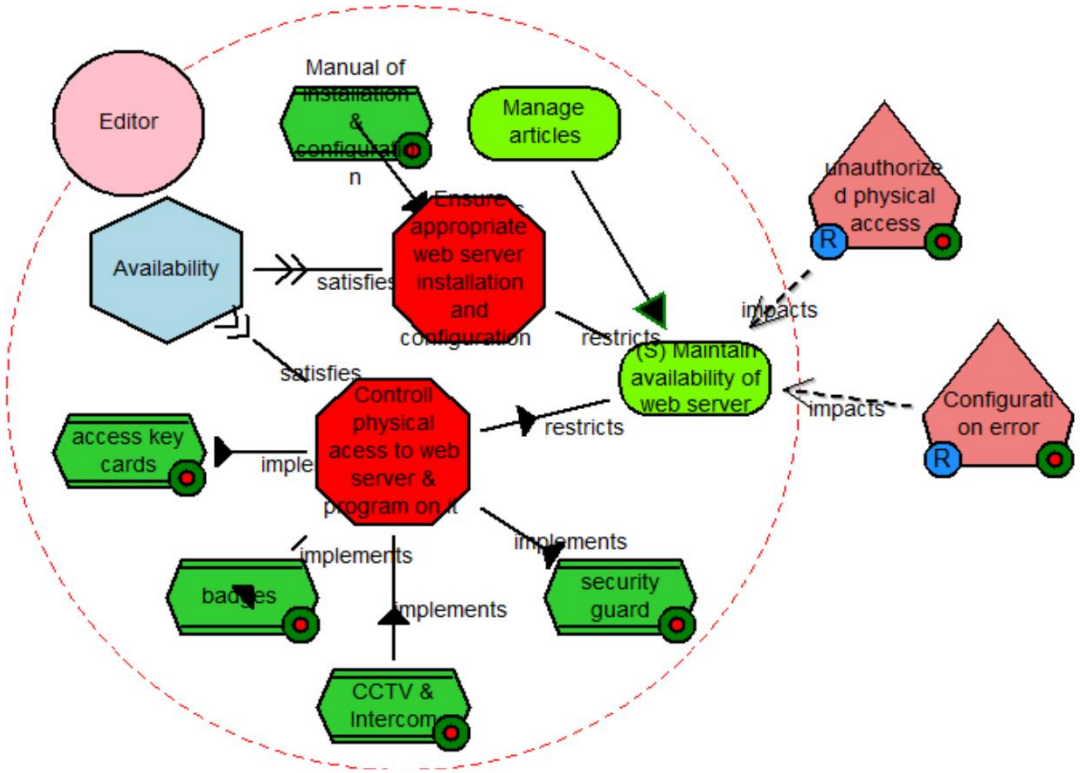


Figure 9.15. Security requirements view (SG1) – publishing domain

- Generated specification

The textual specification is the other output of AMAN-DA. Figure 9.16 represents part of the specification generated, it contains the name of the organization, the different asset to protect, the potential security requirements to consider.

Organization: Publishing Company

Asset: web server

Security Criterion: availability

Req1. The web publishing system should prevent unauthorized acess.

Req2. The editor should ensure appropriate web server's configuration and installation

Req3. The editor should control physical acess to the web server and the program on it.

Asset: submitted articles

Security Criterion: integrity

Req1. The web publishing system should lock editor, reviewer and author, accounts after defined number of login attempts.

Req2. The web publishing system should ask for renaming administrator accounts.

Figure 9.16. Textual specification– publishing domain

6.2. APPLICATION OF AMAN-DA WITH A SALES DOMAIN ONTOLOGY

The following application was adapted from a case study presented in (Mayer, 2012).

- *Inputs of AMAN-DA*

In addition to the core security ontology, figure 9.17 presents the input ontology, a sales domain ontology that encapsulates some knowledge related to the sales domain. One can see the organisation (sales company), which has the agents (Study office, client and sales department); the objects (calculations, technical plans, and estimates); the actions required and performed by the different agents (calculate structure, provide estimates)

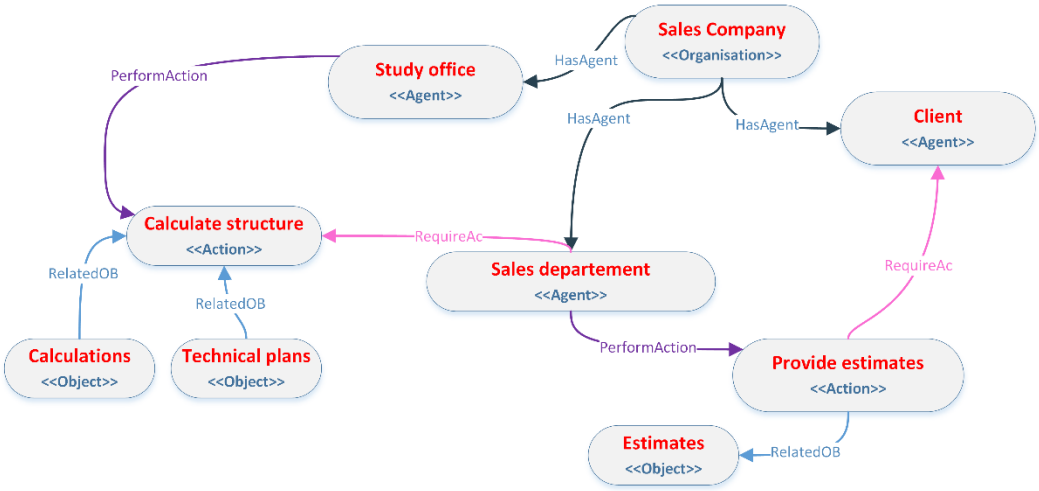


Figure 9.17. Sales domain ontology

This is the security goal captured from stakeholders and formalized according to AMAN-DA:

SG. Keep the integrity of calculations.

- *Outputs of AMAN-DA*

Through the use of AMAN-DA, the security requirements (Req_i) that materialize the input security goals (SG) are defined. The following corresponding Secure Tropos models are generated as well as the final specification.

- Generated Secure Tropos models

Figures 9.18 and 9.19 represent the Secure Tropos models generated after applying AMAN-DA to the previous security goal (SG). Figure 9.18 presents part of the organizational view generated. It contains the main actors (Study office, Sales departments, Clients) and the goal dependencies between them as well as the name of the organization (Sales company).

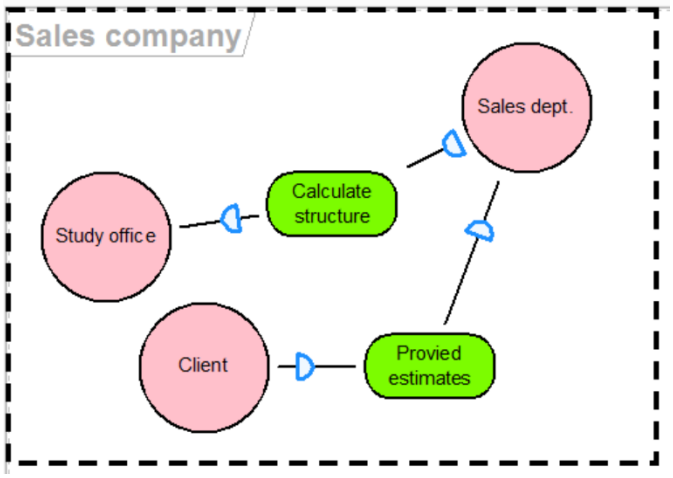


Figure 9.18.Organisational view – Sales domain

Figure 9.19 represents part of the security requirements view that corresponds to the security goal (SG). This view includes the main security requirements to consider (security constraints in Secure Tropos)

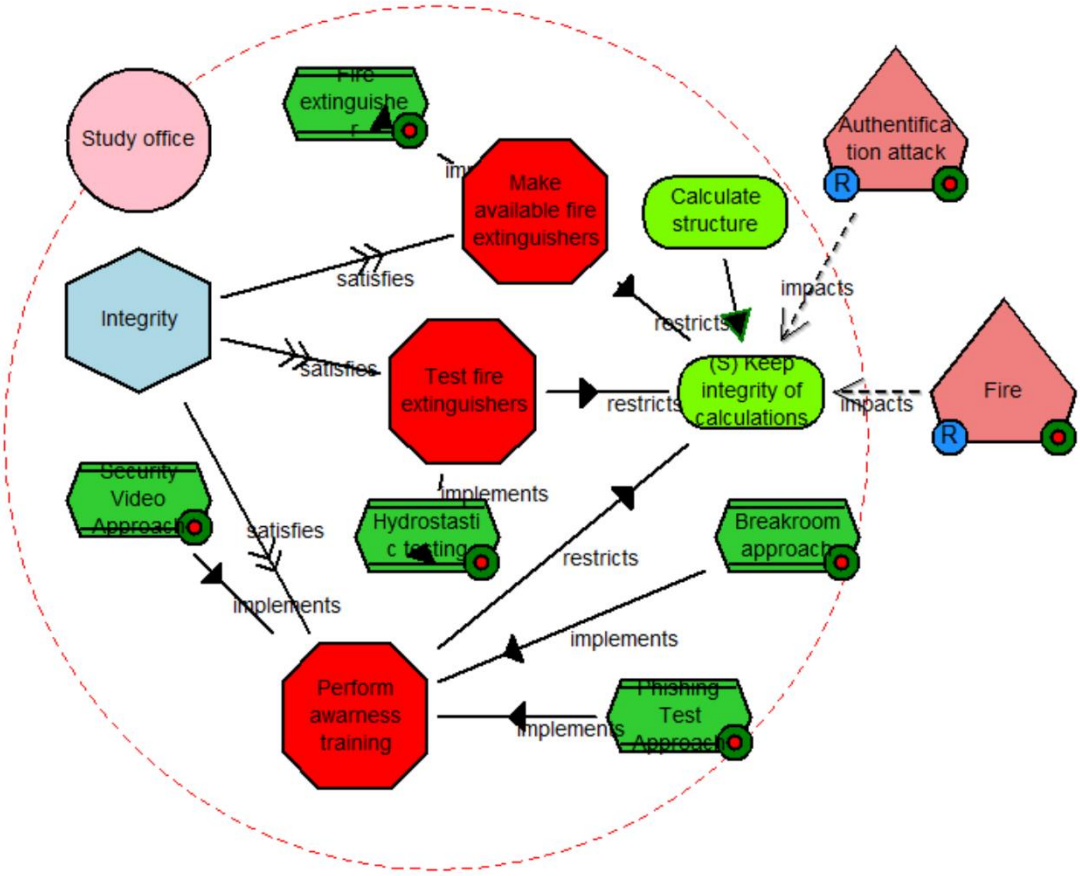


Figure 9.19. Security requirements view – Sales domain

- Generated specification

The textual specification is the other output of AMAN-DA. Figure 9.20 represents part of the specification generated; it contains the name of the organization, the different assets to protect, and the potential security requirements to consider.

Organization: Sales Company

Asset: calculations

Security Criterion: integrity

Req1. The study office should perform awareness training.
Req2. The study office should test fire extinguishers.
Req3. The study office should make a fire extinguisher available.

Figure 9.20. Textual specification – Sales domain

6.3. GENERALISATION

In this thesis AMAN-DA has been applied to four domains (D1: online shopping, D2: maritime, D3: online publishing and D4: sales). With each one, AMAN-DA produced domain specific security requirements. Can we say that what is true for the domains D1, D2, D3 and D4 can be true for all domains? Now, all domains seem to share high level concepts and relations covered by the upper level of the multi-level domain ontology presented in chapter 7. So to reply to the question: *Can AMAN-DA be applied to any domain?*

The answer is yes, if and only if: (i) this domain can be represented with the multi-level domain ontology. (ii)The security goals are formalized according to AMAN-DA. (iii)The core security ontology is used.

6 CONCLUSION

This chapter presented first the application of AMAN-DA to a real world case study. The case study (the security goals) was elaborated with experts from the maritime domain. The models and specification generated demonstrate the ability of AMAN-DA to produce a domain specific security requirements engineering analysis. The chapter examines the application of AMAN-DA to more than one domain (through two other domain ontologies). This latter shows that AMAN-DA is applicable to any domain as long as this domain respects AMAN-DA characteristics. These are promising contribution that need to be reinforced by experimenting with AMAN-DA and this case study with end users.

Chapter 10

Controlled experiment

Any proposed method remains incomplete until its usability and benefits are evaluated by its end-users. Controlled experiments are one way to evaluate methods. During experiments, evaluators need to follow a thorough process. The process explicates the objectives of the evaluation, the different hypotheses to test, the subjects that perform the experiments, and the different variables to measure.

This chapter evaluates the potential practical benefits for end-users that one can expect from the use of AMAN-DA.

1 EVALUATION METHOD USED

Evaluating a method is not an easy task. The evaluation of AMAN-DA was performed based on a selection of evaluation methods in literature.

Note that, despite its importance in methods' life, very little research tackles the topic of the evaluation of methods. Most literature deals with evaluating information systems and software. In (Moody, 2003), the author argues that theoretical models used to explain and predict user acceptance of information technology may be adapted to explain and predict the adoption of methods.

The evaluation process carried out to evaluate the method was adapted from (Pfleeger, 1995). It contains four main steps (Objectives, Design, Protocol and Results) as depicted in Figure 10.1.

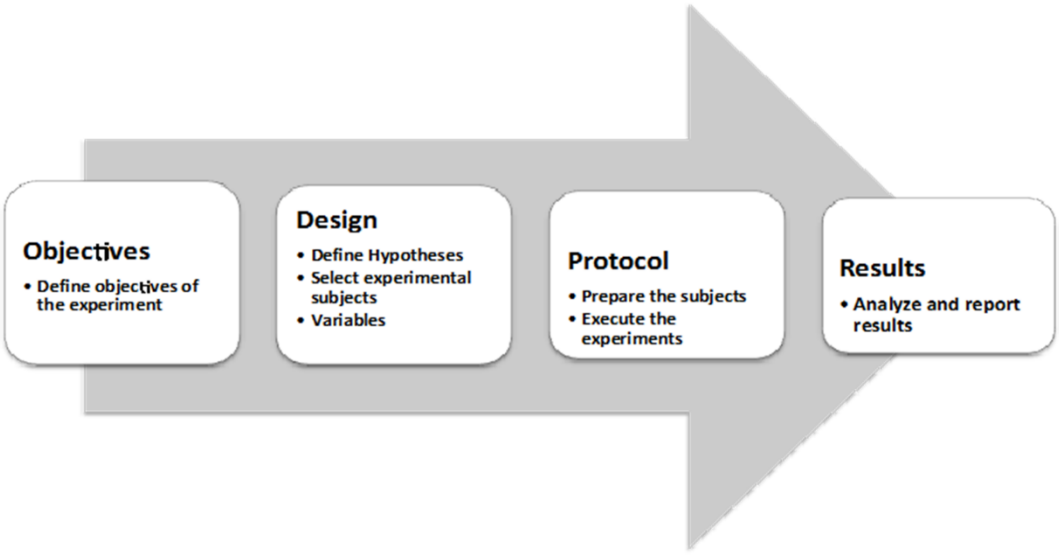


Figure 10.1. Evaluation process adapted from (Pfleeger 1995)

First, the objectives of the evaluation must be defined. The design step consists of defining the hypotheses to test; selecting the experimental subjects (the users that will evaluate the method); and stating and describing the variables to measure. The protocol describes how the experiments were actually performed, starting by preparing the subjects (explaining the method and its tool) and executing the experiments. Finally, the results of the experiment are analyzed and reported.

In (Moody, 2003), the author proposes a framework to evaluate IS design methods based on the work of (Davis, 1989). Based on this framework, we prepared a template (measurement variables and their corresponding questions) to evaluate the method, according to different aspects that were embodied into variables:

- ***Efficiency*** is the effort required to apply a method.
- ***Perceived usefulness*** is defined as "the degree to which a person believes that using a particular system would enhance his or her job performance."
- ***Perceived ease of use*** refers to "the degree to which a person believes that using a particular system would be free of effort."
- ***Intention to Use***: "the extent to which a person intends to use a particular system"

Throughout the evaluation process, the guidelines for preceding context, design, analysis, data collection and reporting and interpreting results proposed by Kitchenham (Kitchenham et al.,

2002) were taken into consideration. These guidelines were useful in, for example, defining the hypotheses, and the selection of the subjects.

The next section presents the evaluation in detail.

2 EVALUATION

2.1. OBJECTIVES: The main goals of experiments were to measure:

- The coverage of the core security ontology and its usability.
- The usage and usability of the method for producing security requirements for specific domains.
- The usability of the tool.

2.2. DESIGN:

A. Hypothesis

Easterbrook et al. (Easterbrook et al., 2008) states that is important to define clear hypotheses before conducting an experiment. The validation of hypotheses will allow the achievement of the evaluation's objectives.

Table 10.1 below summarizes the hypothesis we want to test and their respective objectives that we want to measure.

Table 10.1. Objectives and Hypotheses

Objective to measure	Hypotheses
The security ontology and its usability	- The security ontology provides the main concepts for security requirements elicitation process. (1) - The security ontology is enough to build Secure Tropos models. (2) - The security ontology is enough for elicitating security requirements for different domains. (3)
Usage of the method (Security ontology + domain ontology)	- Using the method for elicitating security requirements makes a difference compared to using only the security ontology (4) - The method is useful for the elicitation of domain specific security requirements. (5)

	- The method is useful for building domain specific Secure Tropos models. (6)
Usability of method	- The method is easy to use (7) - The method is efficient. (8)
Usability of the tool	- Users find the graphical interface to access the security ontology easy to use. (9) - Users find that the tool is easy to use overall. (10) - Users find that the tool is efficient. (11)

B. Subjects

In order to obtain a representative group of subjects (Kitchenham et al., 2002), we contacted, by mail and phone, people from requirements engineering and security communities (laboratories, associations, LinkedIn…). People not related to the field (industrials or researchers) were intentionally excluded. We used the profile page and the job position to include/exclude a subject.

The experiment has been conducted on a group composed of twelve participants. Their average age was 31 years old (min. 25, max. 38). 7 of them were women, 5 men. Among the participants there were PhD students chosen based on their subjects of research that were related to our research area. There were research and teaching staff, and three trainee engineers.

In addition to that, three participants were certified ISO27000, and three had industrial experience with EBIOS (EBIOS, 2004) (a well-known French risk assessment method). The chosen PhD students were working on subjects related to our research area.

Table 10.2 summarizes participants' characteristics.

Table 10.2. Profile of the participants

Number	Age	Sex	Functions
12	31 years old (Min. 25, max 38)	7 women, 5 men	5 PhD students, 4 research and teaching staff, 3 trainee engineers.

B. Variables

Throughout the experiment, the focus was on the usability of the core security ontology, the usage and usability of the method, and the usability of the tool.

The *usability* is defined by ISO 9241 as "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use."[16]. We measured the usability based on the three variables proposed by Moody et al. (Moody, 2003), described in the beginning of the chapter: *ease of use*, *efficiency*, *intention to use*.

The *usage* is defined as "The action of using something or the fact of being used"[17]; for the usage, the focus was on the variable: *usefulness.* (Moody, 2003)

C. Protocol

A mini-workshop was organized for the experiment. During the intended day, 10 participants were present. Two of the participants had impediments, so we booked another time with them and they performed the experiment separately.

The experiment was divided into 5 phases:

- **In phase 1:** participants were welcomed, the aim of the experiment was presented, and the timing, consent to participate and confidentiality were agreed on. The participants affirmed their consent to participate in the experiment by filling out the questionnaire. We affirmed the anonymity of the questionnaire as well as the results of the experiment. The participants were then asked to fill out some pre-evaluation questions related to their age, sex, position and sector of activity in addition to their daily practices related to security requirements engineering.
- **In phase 2:** The core security ontology was presented. Its dimensions, concepts, relations and instances were described in detail. This was followed by discussion with participants. They were then asked to fill in the questionnaire (cf. Appendix F).
- **In phase 3 and phase 4:** A presentation about Secure Tropos was made, mainly to participants who did not know it before. Then, the method was presented through an illustrative example, with its inputs and outputs and its production rules. The method

[16] http://www.w3.org/2002/Talks/0104-usabilityprocess/slide3-0.html
[17] http://www.oxforddictionaries.com/definition/english/usage

was also presented through its tool for a better visibility and comprehension. This was also followed by discussions with participants and they were asked to fill in another part of the questionnaire. In phase 3, the questions concentrated on the usage of the method. Phase 4's questions were related to the usability of the method.

- **In phase 5:** participants were invited to manipulate the tool that implements the method and the core security ontology. They produced their own Secure Tropos models and specifications. They were asked after that to evaluate the tool by replying to the last part of the questionnaire.

Each series of questions in the questionnaire is intended to test one or more hypotheses and evaluate one or more variables. Some questions were open questions; some others were more satisfaction-related, with participants asked to grade their degree of agreement towards a question.

The next sections describe the results obtained for each hypothesis tested.

3 RESULTS

A. The core security ontology and its usability

The items in figure 10.2 allowed us to evaluate the usability of the security ontology with subjects.

Q10. Do you find that the security ontology have the main concepts for security requirements elicitation?
Q11. Does the security ontology help in finding new elements (security requirements, threats, vulnerabilities)?
Q12. Does the security ontology help for building Secure Tropos models?
Q13. Does the security ontology help for defining security requirements for specific domains?

Figure 10.2. Items about the usability of the security ontology

The results extracted from these questionnaires are summarised in Tables 10.3, 10.4, 10.5 and 10.6.

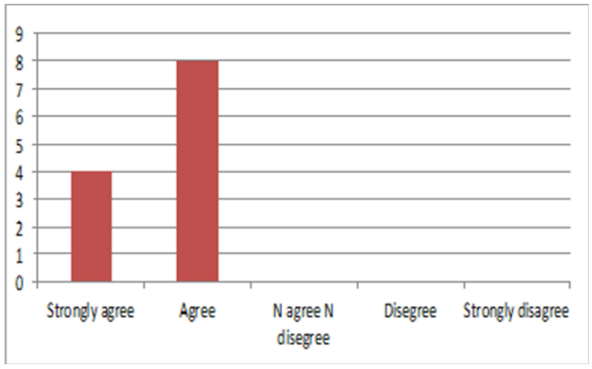


Table 10.3. Results of Q10:

Do you find that the security ontology has the main concepts for security requirements elicitation?

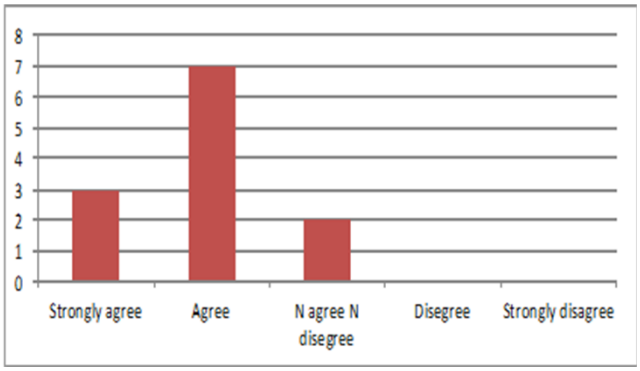


Table 10.4. Results of Q11:

Does the security ontology help in finding new elements (security requirements, threats, and vulnerabilities)?

Tables 10.3 and 10.4 show a quite high level of satisfaction, which is encouraging. Most participants find that the security ontology includes the main concepts. Among the positive qualitative feedbacks that were provided by participants: "***The ontology helps in discovering new elements even for those who are experts in security since it is not easy to have in mind hundreds of threats, vulnerabilities, and their corresponding security requirements***".

According to one participant (a security expert) "***I find in the ontology all concepts that are used in risk analysis methods such as EBIOS***".

One participant mentioned that: "***The ontology seems to have main concepts and individuals, however it would be nice to update it constantly, there are new threats appearing every day!"*** That was an interesting point that could be improved in the future by providing a mechanism to update the individuals of the security ontology automatically. Hypothesis (1) is then mostly validated, which means that the proposed security ontology provides the most important concepts for security requirements elicitation.

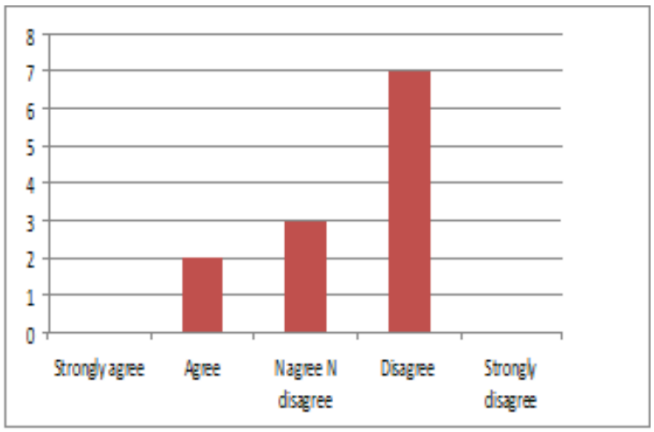


Table 10.5. Results of Q12:

Does the security ontology help for building Secure Tropos models?

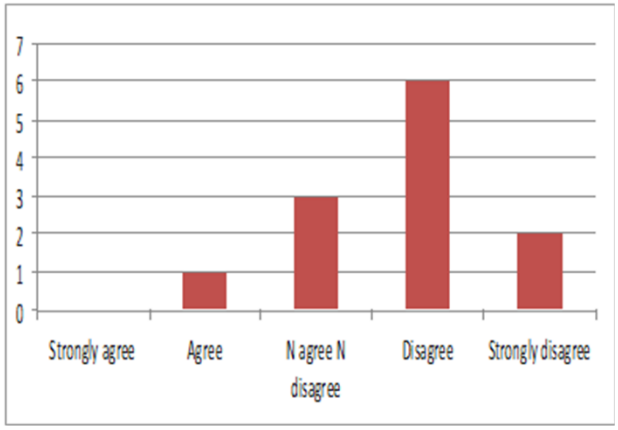


Table 10.6. Results of Q13:

Does the security ontology help for defining security requirements for specific domains?

Questions Q12 and Q13 were devoted to comparing the use of the ontology by itself without the proposed method. The participants were asked: (Q12) Does the core security ontology help in building security models? (Secure Tropos models were taken as an example of a security-modeling framework).

Table 10.5 reports the results for question (Q12), most participants find it difficult to pass from the concepts of the core security ontology to the concepts of Secure Tropos. A common answer was: "***We understand the existence of connections but the mapping from the core security ontology to Secure Tropos is not straightforward***". The discussion with participants that followed this question shows that, although the security ontology has the main concepts, relations and individuals, it remains not enough for users to build security models with it. More guidelines or mapping rules are necessary, not in the ontology itself but in the process using it for security requirements elicitation. Hypothesis (2) is not validated, which means that the security ontology by itself is not enough to build Secure Tropos models.

The last question was: (Q13) does the security ontology help in eliciting security requirements for other specific domains (health, military, and bank)? We wanted to know if the security ontology helps in providing more security domain specific knowledge each time one switches from a domain to another one.

Table 10.6 reports results for question (Q13) and shows that most participants “disagree” on the fact that the security ontology by itself is sufficient for eliciting security requirements for

different specific domains. One participant mentioned: "***This ontology is generic and by definition not domain specific***, ***something additional is required for the application to different specific domains***". This allows us to say that hypothesis (3) is not validated.

The participants started discussing how to make the ontology more domain specific by collaborating with domain experts and with consulting the documentation of each domain.

Participants were wondering if this issue could be made methodological and automatic. That is what we presented for them during the phase 3 and 4.

B. Usage of the method

The items in Figure 10.3 allowed us to evaluate the usage of the method with subjects.

Figure 10.3. Items about the usage of the method

Q14. Do you think the method makes an improvement in the elicitation of security requirements for specific domains?
Q15. Do you think that the method will be effective in discovering new security requirements for the specific domain compared to other methods?
Q16. Do you think the method is useful overall?

The results extracted from these questionnaire are summed up in Tables 10.7, 10.8, and 10.9

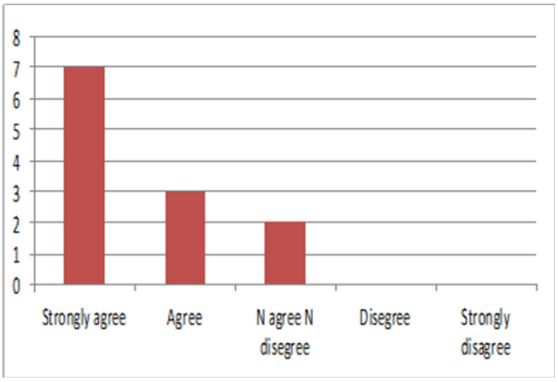


Table 10.7. Results of Q14:

Do you think the method makes an improvement in the elicitation of security requirements for specific domains?

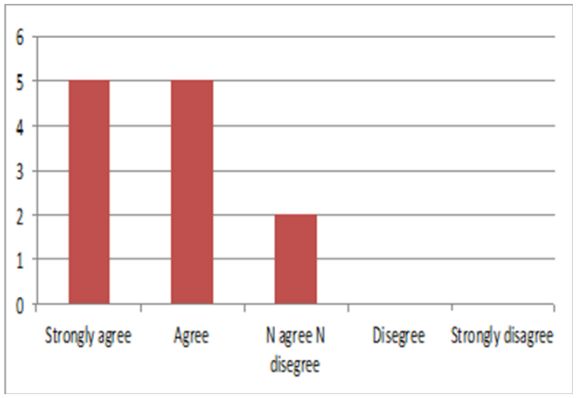


Table 10.8. Results of Q15:

Do you think that the method will be effective in discovering new security requirements for the specific domain comparing to other methods?

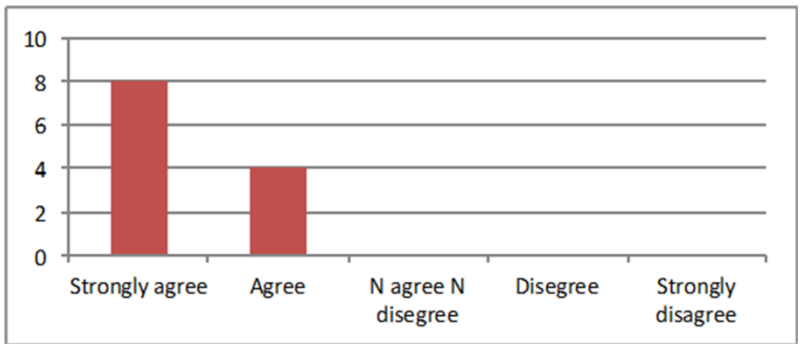


Table 10.9. Results of Q16:

Do you think the method is useful overall?

Table 10.7 and Table 10.8 report the results of participants' answers to Q14 and Q15. The results express "agreement" and "strong agreement" of the participants regarding the advantages of the method for the elicitation of security requirements for specific domains, first compared to the use of the core security ontology only, and second compared to their previous practices using other methods. One participant view was: "***This is definitely better, provided that the domain specific ontology is adequate***". Another participant mentioned: "***Yes, this is better because many specific threats/vulnerabilities, and requirements are listed for each asset.***"

All participants agreed on the usefulness of the proposed method (Table 10.9, Q116), "***In particular with regard to traditional risk assessment methods like EBIOS***" according to a participant.

Hypotheses (4) and (5) are validated. That means that using the method for eliciting security requirements makes a difference compared to using the security ontology only. Also, the method is useful for the elicitation of domain specific security requirements.

C. Usability of the method

The items in Figure 10.4 allowed us to evaluate the usability of the method with subjects.

Q17. Do you think that the method reduces the effort for eliciting security requirements?
Q18. Do you think that the method reduces the effort required to build security requirements models for different domains?
Q19. Do you think the method is easy to use overall?
Q20. Do you have any further criticisms or modifications to suggest that improve the method?

Figure 10.4. Items about the usability of the method

The results extracted from these questionnaires are summed up in Tables 10.10, 10.11, and 10.12.

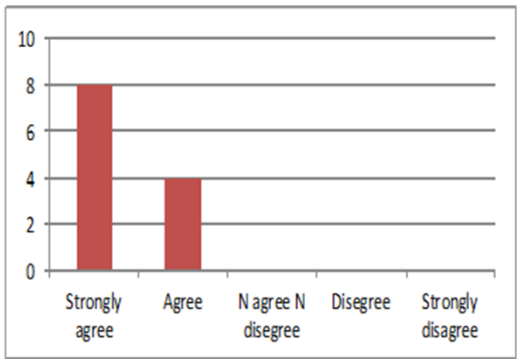


Table 10.10. Results of Q17:

Do you think that the method reduces the effort for eliciting security requirements?

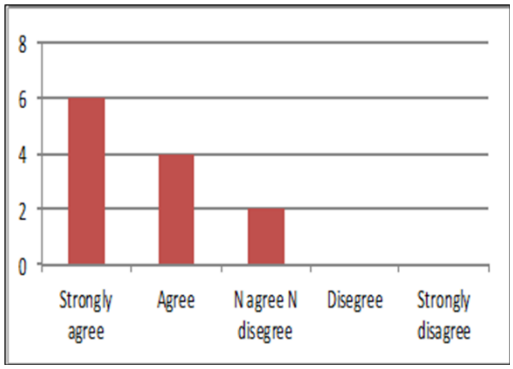


Table 10.11. Results of Q18:

Do you think that the method reduces the effort required to build security requirements models for different domains?

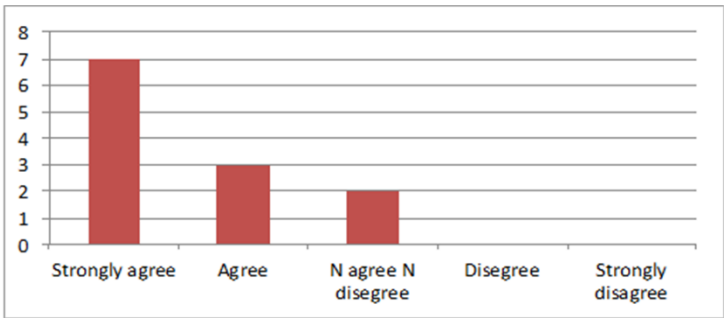


Table 10.12. Results of Q19.

Q17 was related to the elicitation of security requirements using the method and Q18 to building security requirements models. Q19 was about the ease of use of the method. Most participants expressed high level of agreement. One participant compared the method with her usual practice using security standards saying: "***Reading through standards like ISO 27000 is time and effort consuming! The reasoning that the method offers provides the right threats, vulnerabilities, and their corresponding security requirements*** ". Users of Secure Tropos realized how much more efficient it is to use the method compared to their usual practice of building Secure Tropos models where only the concepts are provided, but no knowledge and no reasoning is available.

However, some participants recommended the generalizing the approach to other modeling frameworks (not just Secure Tropos), which was a considerable point. One participant suggested using a verb ontology during the formalization of security goals. (Recall that a security goal is a combination of a verb, one or more security criteria, and one or more assets.) She suggested that it would be nice to have verbs that are more plausible in security such as: keep, maintain, protect, save…

These results validate hypotheses (7) and (8). That implies that the method is easy to use and efficient with a high level of confidence.

D. Usability of the tool implementing the method and the core security ontology

The items in figure 10.5 allowed us to evaluate the usability of the tool implementing the method and the core security ontology with subjects.

Q21. Do you find the interface to access to the core security ontology easy to use?
Q22. Is this prototype friendly (clear and easy) to use?
Q23. Would you need help during the use of the prototype?
Q24. Would you use the prototype in other projects? Which ones?

Figure 10.5. Items about the usability of the prototype implementing the method

The results extracted from these questionnaires are summed up in Tables 10.13, 10.14, and 10.15.

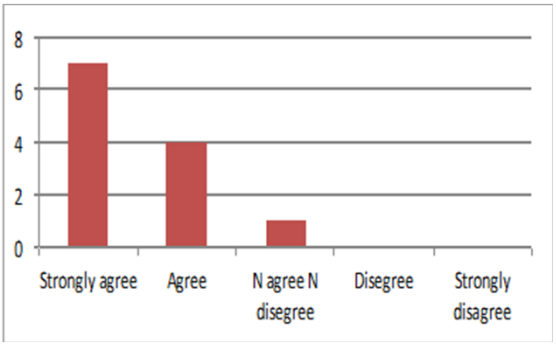


Table 10.13. Results of Q21:

Do you find the interface to access to the core security ontology easy to use?

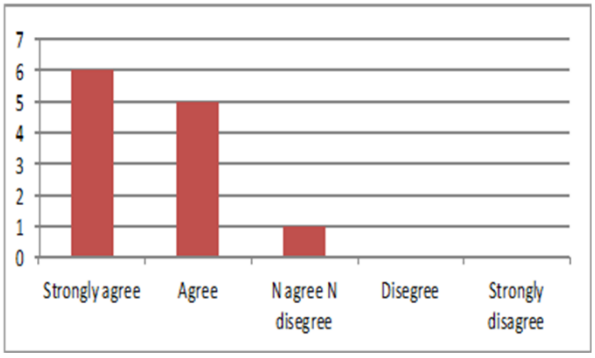


Table 10.14. Results of Q22:

Is this prototype friendly (clear and easy) to use?

As the tool gives access to the core security ontology, almost all participants appreciated the interactive environment that allows this access; they revealed that is nice to have the code of the ontology (in OWL-Protégé) hidden.

Most participants agreed that the tool implementing the method is friendly and easy to use; some of them suggested using more standard interfaces, and to put only one step per window.

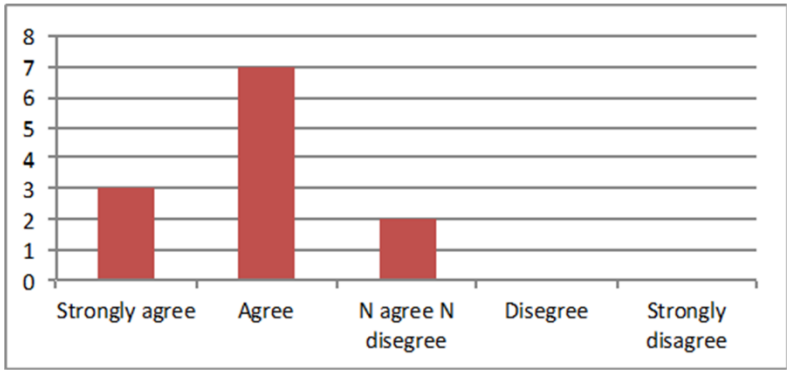


Table 10.15. Results of Q23 :

Would you need help during the use of the prototype?

Surprisingly, as table 10.15 reports, most participants replied that they needed help to use the tool, which was quite disappointing. However when discussing with them, they emphasis that the help is needed only for the first time using the tool. Participants suggested adding a help menu and a guide user document to avoid the misuse of the tool. This was a fair point that we had not thought of, and will be added to the next versions of the tool.

To Q24, plenty of participants expressed their intention to use the tool in other projects. One participant (academic and teacher) considered using the tool in teaching projects, case studies given to his students in security classes.

These results validates hypotheses (9) and (11) which means that users find the graphical interface to access the security ontology easy to use and the tool efficient. Hypothesis (10) is partially validated, which implies that some improvements should be provided to the tool.

4 Summary

Table 10.16 presents the synthesis of the objectives we wanted to measure, their corresponding hypotheses and the main feedback recorded from participants.

Table 10.16. Summary

Objectives to measure	**Hypotheses**	**Feedback of subjects**
The security ontology and its usability	The security ontology provides the main concepts for security requirements elicitation process. (1) The security ontology is enough to build Secure Tropos models. (2) The security ontology is enough for eliciting security requirements for different domains. (3)	The ontology covers wide range of concepts, but should be updated constantly with new elements (threats, vulnerabilities, requirements, etc.) The security ontology by itself is not enough to build Secure Tropos models.
Usage of the method (Security ontology + domain ontology)	Using the method for elicitating security requirements makes a difference regarding using the security ontology only (4) The method is useful for the elicitation of domain specific security requirements. (5) The method is useful to build domain specific Secure Tropos models. (6)	The method makes domain specific security requirements elicitation better provided that the domain ontology is adequate. Specific vulnerabilities/threats are listed for each domain asset.
Usability of method	The method is easy to use (7) The method is efficient. (8)	The method is time saving and easy to use. It is better than reading through ISO27000 standards.
Usability of the tool	Users find that the graphical interface to access the security ontology easy to use. (9) Users find that the tool is easy to use overall. (10) Users find that the tool is efficient. (11)	The interface is friendly and easy to use. The tool however, needs help to be used, mainly for the first time.

The method and its tool were developed to assist and guide users in the elicitation of security requirements during early stages of systems developments – a task known to be difficult due to the tacit, informal knowledge of these users about security and the domain at hand. This

step is often left to ad hoc practices such as copy pasting security requirements from other projects. It is even harder when it comes to building security requirements models. Users left to the concepts of the language (Secure Tropos for example) cannot do much.

Our method and its tool meet users' needs (as the results of the experiment demonstrate). The developed core security ontology, the reasoning rules that we defined to elicit and build Secure Tropos models, and the mechanisms we use to make these requirements more domain specific via the use of different domain ontologies, all make it possible to obtain a structured and structuring outcome.

Many ideas were proposed by the participants of the experiment, to improve the method and to make the tool more attractive.

5 THREATS TO VALIDITY

Controlled experiments are a very good way to validate methods. However, some bias may occur during their implementation. In our case, some threats to validity might have been present. The following points summarize them:

- Despite our efforts to gather a big group of subjects through contact by mail and phone and taking appointments, the number of participants was relatively small (12 persons) due to various impediments. We would like to try the method in a larger group during bigger workshops.
- Some of the participants in the experiment (5 participants) are members of the same laboratory, which might have biased the results.
- The choice of the students to participate in the experiment was based on their field of studies that is related to security and requirements engineering. That was good but might not be enough, since these students lack real world experience and their understanding of software security and systems requirements can be quite peripheral and patchy.

6 CONCLUSION

This chapter reported a controlled experiment performed with a group of users to evaluate the usage and usability of AMAN-DA, the coverage of the core security ontology and the

usability of the AMAN-DATool. This experiment revealed that the core security ontology covers the main security concepts and helps in discovering new security elements that one might not think of. However it can and should be updated constantly with new threats and vulnerabilities. The experiment revealed also that the method is beneficial for the elicitation of domain specific security requirements. Finally, users involved in the experiments find the tool friendly and easy to use. The discussion with users at the end of the experiment indicated new questions and possible improvements of the method; those will be reported as items for possible future research in the conclusion of this thesis.

Conclusions and future research

The main objective of this thesis is to answer the research question: *How to elicit security requirements that are domain specific with a reuse strategy?*

To answer this question, this thesis proposes:

- A framework for analyzing and comparing knowledge reuse in SRE. The framework was defnied after a systematic mapping study of knowledge reuse in security requirements engineering.
- An original framework to classify security ontologies.
- A core security ontology.
- A multi-level domain ontology to structure domain knowledge.
- Linguistic templates to specify security goals and security requirements.
- A rule-based mechanism to extract relevant knowledge from domain and security ontologies and produce security requirements models and textual security requirements specifications.
- A tool to elicit security requirements and implement the proposed method.
- An evaluation through a real case study and a controlled experiment with end users.

The work presented in this PhD has been proven to be interesting and beneficial for the requirements engineering and security communities. This has been noticed during the evaluation phase. The work has been also reported and presented during different national and international workshops, conferences and journals. However, like any research work, there is always room for improvements, new research perspectives. The systematic mapping study on knowledge reuse in requirements engineering should be constantly updated. The SMAP revealed many issues related to lack of automation and standardisation in SRE methods. There is a lack of domain specific security requirements engineering. One main possible follow-up of the SMAP is to invite the research community to collaborate more to propose some kind of unification (like in UML) of the different methods proposed for security requirements engineering taking in consideration a reuse strategy and domain specific analysis. Many security ontologies have been proposed by the research community but, as

shown by the literature survey, none of them covers all aspects of security. This thesis has proposed a core security ontology based on the available security ontologies existing in the literature, which is a step forward. However, this core security ontology needs to be updated constantly and automatically with new threats and vulnerabilities. Again, a call for standardisation seems appropriate: the community needs a good standard security ontology, possibly one for each domain. The one proposed in this thesis can be a good beginning for such a project.

AMAN-DA, the method proposed in this thesis, has many benefits for domains specific security requirements elicitation. The syntactic requirements patterns for documenting security goals and security patterns can be improved and compared to other patterns existing in the literature. An open question would be, what are the necessary parameters to take in consideration in the formulation of the security goals and security requirements? For instance, the level of security (i.e. low, medium, high) is an example to explore.

It would be nice to extend AMAN-DA to be used with other languages (and not only Secure Tropos), but this perspective is pending until we have final and standard versions of the different languages meta-models. The Secure Tropos meta-model has already gone through more than three versions since this PhD started, and that is almost the case for the other languages (such as I*, KAOS, …)..

In this PhD thesis, we chose to use a security ontology and a domain ontology at the same time. Another strategy that arouses our curiosity is to try a strategy where a domain specific security ontology is developed for each domain, then used for security requirements elicitation based on the same idea of rules. A typical requirements engineering process includes requirements elicitation, analysis, verification and validation. AMAN-DA handles so far the elicitation and the analysis steps, it would be interesting to have an advanced version of AMAN-DA that handles the two other steps of the verification and validation of the security requirements.

Using domain ontologies through AMAN-DA is useful, but one shouldn’t neglect the fact that the domain ontologies covers the domain knowledge related to the domain at hand and part of it can be very sensitive piece of domain information, for example the submitted articles in the web publishing domain ontology (Figure 9.13) or the crew evaluation in the maritime domain ontology (Figure 7.3). AMAN-DA should in the long-term future ensure the possibility to secure the domain information that we don’t want to be public.

The AMAN-DATool developed can be improved, its interface can be made more friendly, a help menu can be added. A project we are currently considering is to merge it with the SecTro Tool to have a knowledge based, domain specific SecTro analysis and elicitation.

Appendices

Appendix A

Systematic Mapping Study: Retrieved publications

Table A.1. (in the following) presents the searches conducted and the list of all publications retrieved in our systematic mapping study. For each paper retrieved, the following information is provided: name of the first author, title of the paper, year of publication and digital library/resource. For each category, and for each conference/workshop, the table gives the number of papers found followed by the number of papers selected (using our selection criteria). The black color refers to papers selected; the green color refers to papers not selected in the SMAP.

Table A.1. Table of all retrieved papers.

First Author	Title	Pub. Year	Digital library/resource
Books & Book Chapters, Phd. Found=18, Selected=10			
Mayer N.	Model-based Management of Information System Security Risk	2012	Amazon
Lund, M. S.	The CORAS Tool	2011	SpringerLink
Hull, E.	Requirements Engineering.	2011	GoogleBooks
Yu, E.	Modelling strategic relationships for process reengineering.	2011	GoogleBooks
Lopez, J.	Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Networks	2009	Foundations of Security Analysis and Design
Massacci, F.	An ontology for secure socio-technical systems.	2007	Handbook of Ontologies for Business Interaction.
Lamsweerde, A.	Engineering Requirements for System Reliability and Security.	2007	IOS press ebooks
Giorgini, P.	Security and Trust Requirements Engineering	2005	Foundations of Security Analysis and Design
Jürjens, J.	Secure systems development with UML.	2005	Amazon
Ivankina, E.	An Approach to Guide Requirement Elicitation by Analysing the Causes and Consequences of Threats	2005	Information Modelling and Knowledge Bases
Kruchten, P.	The Rational Unified Process: An Introduction	2004	Amazon
Jackson, M. J	Problem Frames: Analysing & Structuring Software Development Problems	2001	Amazon
Yu, E.	Modelling Trust for System Design Using the i * Strategic Actors Framework	2001	SpringerLink
Antón, A	Strategies for Developing Policies and Requirements for Secure Electronic Commerce Systems.	2000	SpringerLink
Jacobson, I.	The unified software development process.	1999	Amazon
Kotonya, G.	Requirements engineering: processes and techniques.	1998	Amazon
Jackson, M. J.	Software requirements & specifications: a lexicon of practice, principles, and prejudices	1995	Amazon
Abiteboul,S.	Foundations of databases	1995	Amazon
Journals, Found=31, Selected=20			
Computer, Found=1, Selected=1			
Nuseibeh,B.	Securing the skies: in requirements we trust	2009	IEEE Computer society
Journal of Electronic Security and Digital Forensics, Found=1, Selected=0			
Ivan,F.	Integrating security and usability into the requirements and design process	2007	ACM
Journal of Software Engineering and Knowledge Engineering, Found=3, Selected=2			
Mouratidis, H.	Modelling Secure Systems Using an Agent-Oriented Approach and Security Patterns.	2006	Google scholar
Mouratidis, H.	Secure Tropos: A Security-Oriented Extension of the Tropos Methodology	2006	Google scholar

Bauer, B	Agent UML: A formalism for specifying multiagent software systems.	2000	citeseerx
Electronic Journal for E-Commerce Tools and Applications. Found=1, Selected=1			
Dritsas, S.	A knowledge-based approach to security requirements for e-health applications	2006	www.ejeta.org
Autonomous Agents and Multi- Agent Systems, Found=1, Selected=1			
Bresciani, P.	Tropos: An agent-oriented software development methodology	2004	SpringerLink
Military Operations Research, Found=1, Selected=1			
Buckshaw, D.	Mission oriented risk and design analysis of critical information systems	2005	ingentaconnect
Security & Privacy, IEEE, Found=1, Selected=1			
Evans, S.	Risk-based systems security engineering: stopping attacks with intention	2004	IEEExPlore
Requirements Engineering Journal, Found=4, Selected=3			
Fabian, B.	A comparison of security requirements engineering methods. Requirements Engineering,	2010	SpringerLink
Sindre, G.	Eliciting security requirements with misuse cases	2005	ACM
Antón, A	A requirements taxonomy for reducing Web site privacy vulnerabilities	2004	SpringerLink
Toval, A.	Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach	2001	citeseerx
Journal of Object Technology, Found=2, Selected=2			
Firesmith,D.	Specifying reusable security requirements.	2004	www.jot.fm
Firesmith,D.	Security use cases.	2003	www.jot.fm
Computers & Security, Found=1, Selected=0			
Gritzalis, D.	Principles and requirements for a secure e-voting system.	2002	sciencedirecte
Software Engineering, IEEE Transactions on. Found=3, Selected=1			
Breaux,T.	Analyzing Regulatory Rules for Privacy and Security Requirements	2008	ACM
Haley, C.B.	Security Requirements Engineering: A Framework for Representation and Analysis.	2008	IEEExPlore
Rolland, C.	Guiding goal modeling using scenarios.	1998	IEEExPlore
Computer Standards & Interfaces. Found=3, Selected=2			
Mellado, D.	A common criteria based security requirements engineering process for the development of secure information systems.	2007	sciencedirecte
Massacci,F.	Using a security requirements engineering methodology in practice: The compliance with the Italian data protection legislation	2005	sciencedirecte
Bhavani,T.	Security standards for the semantic web	2005	sciencedirecte
Computer Communications. Found=1, Selected=0			
Lambrinoudakis, C.	Security requirements for e-government services: a methodological approach for developing a common PKI-based security policy.	2003	sciencedirecte
International Journal of Computer Applications. Found=1, Selected=1			
Salini, P.	A Knowledge-oriented Approach to Security Requirements for an E-Voting System.	2012	www.ijcaonline.org
Informatical journal. Found=1, Selected=1			
Susi, A.	The tropos metamodel and its use.	2005	http://www.troposproject.org
International Journal of Information Security. Found=1, Selected=1			
Giorgini, P.	Requirements Engineering for Trust Management: Model, Methodology, and Reasoning.	2006	IEEExPlore
Journal of systems and software. Found=1, Selected=1			
Mouratidis H.	A framework to support selection of cloud providers based on security and privacy requirements.	2013	Sciencedirecte
Journal of Research and Practice in Information Technology. Found=1, Selected=1			
Velasco, J.	Modelling Reusable Security Requirements Based on an Ontology Framework	2009	Google scholar
Information and Software Technology. Found=1, Selected=0			
Maamar, Z.	Towards an ontology-based approach for specifying and securing Web services	2006	sciencedirecte
Journal of Autonomous Agents and Multi-Agent Systems. Found=1, Selected=0			
Kaga,L.	Modeling Conversation Policies using Permissions and Obligations	2005	ACM
Engineering Applications of Artificial Intelligence. Found=1, Selected=0			
Tan,J.J.	Dynamic security reconfiguration for the semantic web	2004	sciencedirecte
Conferences Found =70, Selected =39			
ARES. Found= 12 , selected selected =6			
Beckers, K.	Comparing Privacy Requirements Engineering Approaches	2012	IEEExPlore
Beckers, K	Using Security Requirements Engineering Approaches to Support ISO 27001 Information Security Management Systems Development and Documentation	2012	IEEExPlore
Karpati, P.	Characterising and Analysing Security Requirements Modelling Initiatives	2011	IEEExPlore
Kárpáti, P.	Experimental Comparison of Misuse Case Maps with Misuse Cases and System Architecture Diagrams for Eliciting Security Vulnerabilities and Mitigations	2011	IEEExPlore
Okubo, T.	Effective Security Impact Analysis with Patterns for Software Enhancement	2011	IEEExPlore

Zuccato, A.	Service Security Requirement Profiles for Telecom: How Software Engineers May Tackle Security.	2011	IEEExPlore
Langer, L.	A Taxonomy Refining the Security Requirements for Electronic Voting: Analyzing Helios as a Proof of Concept	2010	IEEE Computer society
Schmidt, H.	Threat- and Risk-Analysis During Early Security Requirements Engineering	2010	IEEExPlore
Hatebur, D.	A Pattern System for Security Requirements Engineering.	2007	IEEExPlore
Asnar,Y.	From Trust to Dependability through Risk Analysis.	2007	IEEExPlore
Mellado, D.	A comparison of the Common Criteria with proposals of information systems security requirements	2006	IEEExPlore
Giorgini, P.	ST-tool: a CASE tool for security requirements engineering.	2005	IEEExPlore
AINA, Found=1, Selected=1			
Tsoumas,B.	Towards an Ontology-based Security Management	2006	IEEExPlore
CAiSE, Found=2 , Selected=2			
Paja, E.	Modelling Security Requirements in Socio-Technical Systems with STS-Tool	2012	Google scholar
Mouratidis, H.	Integrating security and systems engineering: Towards the modelling of secure information systems.	2003	Citeseerx
COMPSAC; Found= 1 , Selected=0			
Elahi,G.	Security Requirements Engineering in the Wild: A Survey of Common Practices	2011	IEEExPlore
CSEE&T. Found= 1 , Selected=1			
Mead, N.R	Security Requirements Engineering for Software Systems: Case Studies in Support of Software Engineering Education	2006	IEEExPlore
ETRICS, Found= 1 , Selected=1			
Hatebur, D.	Security Engineering Using Problem Frames.	2006	SpringerLink
EEE. Found= 1 , Selected=1			
Marti, R.	Security specification and implementation for mobile e-health services.	2004	IEEExPlore
ENASE. Found=1 , Selected=0			
Semmak, F.	Extended Kaos to Support Variability for Goal Oriented Requirements Reuse	2010	SpringerLink
FIRA - STA. Found=1, Selected=1			
Chikh, A.	An Ontology Based Information Security Requirements Engineering Framework	2011	SpringerLink
HICSS. Found=1, Selected=0			
Goluch, G.	Integration of an Ontological Information Security Concept in Risk-Aware Business Process Management	2008	IEEExPlore
ICSE. Found= 3, Selected =3			
Best, B.,	Model-Based Security Engineering of Distributed Information Systems Using UMLsec.	2007	IEEExPlore
Firesmith,D.	Engineering Safety and Security Related Requirements for Software Intensive Systems.	2007	IEEExPlore
Van Lamsweed	Elaborating security requirements by construction of intentional anti-models.	2004	IEEExPlore
ICSOC. Found=1, Selected=0			
Deubler,M.	Sound development of secure service-based systems	2004	Citeseer
ICICS. Found=1 , Selected=1			
Jensen, J.	Experimental Threat Model Reuse with Misuse Case Diagrams.	2010	SpringerLink
IFIP TC9/WG9.6, Found=2, Selected=1			
Tsoumas,B.	Security by ontology; A knowledge centric approach	2006	SpringerLink
Rannenberg, K.	Recent Development in Information Technology Security Evaluation-The Need for Evaluation Criteria for Multilateral Security.	1993	ACM
iTrust. Found=1, Selected=1			
Vraalsen, F.	The CORAS Tool for Security Risk Analysis.	2005	SpringerLink
MoDELS, Found=2, Selected=2			
Saeki, M.	Security Requirements Elicitation Using Method Weaving and Common Criteria	2009	SpringerLink
Hogganvik, I.	A graphical approach to risk identification, motivated by empirical investigations.	2006	SpringerLink
NIK. Found=1, Selected=1			
Sindre, G.	Capturing security requirements through misuse cases.	2001	Google scholar
RE. Found =24, Selected=6			
Morali, A.	Risk-based Confidentiality Requirements Specification for Outsourced IT Systems	2012	IEEE Computer society
Paja, E.	STS-tool: Socio-technical Security Requirements through social commitments	2012	IEEExPlore
Supakkul, S.	An NFR Pattern Approach to Dealing with NFRs	2010	IEEExPlore
Eunsuk,K.	Dependability Arguments with Trusted Bases	2010	IEEExPlore
Ameller, D.	Dealing with Non-Functional Requirements in Model-Driven Development	2010	IEEExPlore
Hill, J.	Creating Safety Requirements Traceability for Assuring and Recertifying Legacy Safety-Critical Systems	2010	IEEExPlore
Xiping,S.	Experiences in Developing Quantifiable NFRs for the Service-Oriented Software Platform	2009	IEEExPlore
Teng, L.	AVT Vector: A Quantitative Security Requirements Evaluation Approach Based on Assets,	2009	IEEExPlore

	Vulnerabilities and Trustworthiness of Environment		
Jureta, I.J.	Revisiting the Core Ontology and Problem in Requirements Engineering	2008	IEEExPlore
David,C.	Balancing Security Requirements and Emotional Requirements in Video Games	2008	IEEExPlore
Pichler, M.	Agile Requirements Engineering for a Social Insurance for Occupational Risks Organization: A Case Study	2006	IEEExPlore
Juan, P. C.	Managing Non-Technical Requirements in COTS Components Selection	2006	IEEExPlore
Gonzalez-Baixauli, B.	Eliciting Non-Functional Requirements Interactions Using the Personal Construct Theory	2006	IEEExPlore
Chisan, J.	Exploring the role of requirements engineering in improving risk management	2005	IEEExPlore
Cohene, T.	Contextual Risk Analysis for Interview Design	2005	IEEExPlore
Kaiya, H.	Identifying Stakeholders and Their Preferences about NFR by Comparing Use Case Diagrams of Several Existing Systems	2004	IEEExPlore
Haley, C.B.	The Effect of Trust Assumptions on the Elaboration of Security Requirements	2004	IEEExPlore
Lin, L.	Using abuse frames to bound the scope of security problems	2004	ACM
Lin, L.	Introducing abuse frames for analysing security requirements.	2003	Citeseerx
Liu, L.	Security and privacy requirements analysis within a social setting.	2003	IEEExPlore
Steve, L.	The journay toward secure systems: Achieving Assurance	2003	IEEExPlore
Ian, A.	Initial industrial experience of misuse cases in trade-off analysis	2002	IEEExPlore
Wojtek,K.	Requirements, Architectures and Risks	2002	IEEE Computer society
Gene, S.	The Hidden Meta-Requirements of Security and Privacy	2001	IEEExPlore
REFSQ. Found=3, Selected=3			
He, Q.	A Framework for Modeling Privacy Requirements in Role Engineering.	2003	http://www4.ncsu.edu
Sindre, G.	A Reuse-Based Approach to Determining Security Requirements.	2003	Citeseerx
Sindre, G.	Templates for Misuse Case Description	2001	Citeseerx
SAFECOMP. Found=1, Selected=1			
Grünbauer, J.	Modelling and Verification of Layered Security Protocols: A Bank Application.	2003	SpringerLink
SKG. Found= 1, Selected=0			
Vorobiev, A.	Security attack ontology for web services	2006	IEEExPlore
Sicherheit. Found=1, Selected=1			
Gürses, S. F	Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation	2006	DBLP
SIN. Found= 1, Selected= 0			
Parkin, S. E.	An Information Security Ontology Incorporating Human-Behavioural Implications	2009	ACM
TrustBus, Found=1, Selected=1			
Pavlidis, M.	Trustworthy Selection of Cloud Providers Based on Security and Privacy Requirements: Justifying Trust Assumptions	2013	IEEE Computer society
TRUST. Found=1, Selected=1			
Vrakas, N.	Privacy Requirements Engineering for Trustworthy e-Government Services.	2010	SpringerLink
TOOLS-PACIFIC. Found=1, Selected=1			
Sindre, G.	Eliciting security requirements by misuse cases	2000	IEEExPlore
UML. Found=2, Selected=2			
Jürjens, J.	Automated verification of UMLsec models for security requirements.	2004	citeseerx
Lodderstedt, T.	SecureUML: A UML-Based Modeling Language for Model-Driven Security	2002	ACM
WWW. Found=1, Selected=1			
Martí, R.	Security in a wireless mobile health care system.	2003	Google scholar
Workshops, Found =21, Selecetd = 11			
ASIACCS. Found=1, Selected=0			
Fenz, S.	Formalizing information security knowledge	2009	Citeseer
CAiSE workshops. Found=1, Selected=0			
Massacci, F.	An Extended Ontology for Security Requirements	2011	SpringerLink
DEXA. Found=1, Selected=1			
Hatebur, D.	A Security Engineering Process based on Patterns.	2007	IEEExPlore
ESORICS. Found=1, Selected=1			
Mellado, D.	Applying a Security Requirements Engineering Process.	2006	SpringerLink
EDOCW. Found=1, Selected=0			
Naufel do Amaral, F.	An ontology based-approach to the formalization of information security polocies	2006	IEEExPlore
EWSA. Found=1, Selected=0			

Schmidt, H.	Preserving Software Quality Characteristics from Requirements Analysis to Architectural Design.	2006	IEEExPlore
MARK. Found=1, Selected=1			
Salinesi, C.	Using the RITA Threats Ontology to Guide Requirements Elicitation: an Empirical Experiment in the Banking Sector.	2008	IEEExPlore
NSPW. Found=1, Selected=0			
Raskin,V.	Ontology in information security: a useful theoretical foundation and methodological tool	2001	ACM
OTM. Found=1, Selected=1			
Daramola, O.	Ontology-Based Support for Security Requirements Specification Process	2012	Springer
RePa. Found=1, Selected=1			
Daramola, O.	Pattern-based security requirements specification using ontologies and boilerplates	2012	IEEExPlore
RISI. Found=1, Selected=0			
Beckers,K.	An Integrated Method for Pattern-Based Elicitation of Legal Requirements Applied to a Cloud Computing Example.	2012	dblp
RHAS. Found= 2, Selected=1			
Firesmith,D.	A Taxonomy of Security-Related Requirements	2005	citeseerx
Lee, S.W.	Engineering Dependability Requirements for Software-intensive Systems through the Definition of a Common Language	2005	Citeseerx
SecSE. Found=1, Selected=0			
Seeger,M. M.	A Comparative Study of Software Security Pattern Classifications	2012	DBLP
SESS. Found=3, Selected=2			
Lee S.W.	Building problem domain ontology from security requirements in regulatory documents	2006	ACM
Haley, C.B.	A framework for security requirements engineering.	2006	open.ac.uk
Mead, N.R.	Security quality requirements engineering (SQUARE) methodology	2005	ACM
SAC. Found=1, Selected=1			
Jürjens, J.	Using UMLsec and goal trees for secure systems development.	2002	ACM
Spattern. Found=1, Selected=1			
Fernandez, E.B.	Measuring the Level of Security Introduced by Security Patterns.	2010	IEEExPlore
UKDU Workshop. Found=1, Selected=1			
Gürses, S.	Multilateral security requirements analysis for preserving privacy in ubiquitous environments	2006	Citeseer
WSCS. Found=1, Selected=0			
Yang, Y.	Towards Semantic Requirement Engineering.	2008	IEEExPlore
Reports, Found = 18 , Selected = 15			
Wenzel,S.	Approach for adaptive security monitor generation	2012	http://www.securechange.eu
MAGERIT v.3	Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información	2012	http://administracionelectronica.gob.es
Travis,C.	Security Requirements Reusability and the SQUARE Methodology.	2010	sei.cmu.edu
Morali, A.	CRAC : Confidentiality risk analysis and IT-architecture comparison of business networks	2009	http://doc.utwente.nl/
Mead, N.R	Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models	2008	Google scholar
Elahi,G.	A Goal Oriented Approach for Modeling and Analyzing Security TradeOffs	2007	citeseerx
Dahl,H.	Structured semantics for the CORAS security risk modelling language.	2007	coras.sourceforge.net/
Hermoye, L.A.	Attack patterns for security requirements engineering	2006	Google scholar
Mylopoulos,J.	Risk Modelling and Reasoning in Goal Models	2006	http://eprints.biblio.unitn.it
Hermoye, L. A.	A Reuse-Based Approach to Security Requirements Engineering	2006	users.ece.utexas.edu
Massacci, F.	Detecting Conflicts between Functional and Security Requirements with Secure Tropos:	2006	Citeseer
Araujo, R.	Design Authorization Systems Using SecureUML.	2005	Google
Lin, L.-C.	Analysing security threats and vulnerabilities using abuse frames.	2003	Google scholar
Firesmith	Common Concepts Underlying Safety, Security, and Survivability Engineering	2003	sei.cmu.edu
MD, N. S. A. S. S. F. G. G. M	Common Criteria for Information Technology Security Evaluation:	2002	Google scholar
Lin, L.	Analyzing security requirements as relationships among strategic actors.	2002	citeseerx
Schmidt, H.	UML Revision Taskforce.	2001	citeseerx
ECMA-271	Extended Commercially Oriented Functionality Class for Security Evaluation,	1999	Google

Appendix B

Table B.1. Core ontology concepts definition using security ontologies and models

	Concepts of the ontology	Mayer et al.	Tsoumas et al.	Herzog et al.	Fenz et al.	Velasco et al.	Dritsas et al.	Karyda et al.	Kim et al.	Undercoffer et al.	Geneiatakis et al.	Denker et al.	Lekhchine et al.	Martimioano et al.
Orgnization dimension	Asset	Business Asset	Asset	Asset	Asset	Asset	Asset	Asset	-	System component	Target	-	Asset	Asset
	Location	-	-	-	Location	-	-	-	-	Location	-	-	-	-
	Organization	-	-	-	Organization	-	-	-	-	-	-	-	Organization	-
	Person	-	Stakeholder	-	Person	-	Stakeholder	Person	-	-	-	-	Person	Supplier
Risk dimension	**Threat**	Threat	Threat	Threat	Threat	Threat	Threat	Threat	-	-	-	-	Threat	Attack
	Vulnerability	Vulnerability	Vulnerability	Vulnerability	Vulnerability	-	Vulnerability	-	-	Means	-	-	Vulnerability	Vulnerability
	Risk	Risk	Risk	-	-	Risk	-	-	-	-	-	-	-	-
	Severity	-	-	-	Attribute	-	-	-	-	-	-	-	-	-
	Impact	Impact	Impact	-	-	Valuation criteria	-	-	-	Consequence	-	-	-	Consequence
	Threat agent	Threat agent	Threat agent	-	Role	-	Attacker	-	-	-	-	-	-	Agent
	Attack method	Attack method	Attack	-	-	-	Deliberate attack	-	-	Intrusion	SIP_attack	-	-	-
	Attack tool	-	-	-	-	-	-	-	-	-	SIP message	-	-	Tool
Treatment dimension	Security goal	-	-	-	-	-	-	-	-	-	-	-	-	-
	Security criterion	Security criterion	-	goal	Security attribute	-	Objective	Security objective	Security objective	-	-	Security notation	Objective mechanism	Acess
	Requirements document	-	Security policy	-	-	-	-	-	-	-	-	-	-	-
	Control	Control	Countermeasure	Countemeasure	Asset	Safeguard	Countermeasure	Countermeasure	Security mechanism/Credential	-	-	Security mechanism/ Credential	Countermeasure/Protocol/Algorithm	Correction
	Security requirement	Security requirement	Control	-	Control	Security requirement	-	-	-	-	-	-	-	-

Appendix C

Interview instrument

(Maritime case study)

The following presents part of the interview instrument that was prepared before meeting the ship's captain.

1. What is the main role of the ship captain?
2. What are the threats that might threaten the ship, is there a document that document them?
3. What are the information that the captain/crew have access to?
4. Who manages the alarm system in the ship, who has access to it?
5. What are the information that the ship needs to provide to access a port?
6. What needs to be protected in a ship (data, software, equipment)?
7. What are the information that needs to be saved?
8. How is the information saving policy in the ship?
9. What are the main goals of the captain to maintain the ship security systems?
10. What are the documents to consult in terms of ship security?

Appendix D

Implementation

AMAN-DA was automated via a tool that the requirements engineer or the security analyst can use during the security requirements analysis and elicitation process. The tool was implemented on Java Eclipse.

1 A TOOL FOR SECURITY REQUIREMENTS ANALYSIS AND ELICITATION (AMAN-DATOOL)

1.1. FUNCTIONAL AND TECHNICAL ARCHITECTURE

Figure D.1 summarizes the main functionalities that the tool offers to requirements engineers (or security analysts). The tool allows: (i) generic security requirements analysis and elicitation through an interface that accesses to core security ontology (presented in chapter 5), (ii) domain specific security requirements analysis and elicitation, (iii) generating Secure Tropos models and natural language specification documents (using the mechanisms presented in chapter 8). Each functionality includes other sub-functionalities that will be presented in details throughout this chapter.

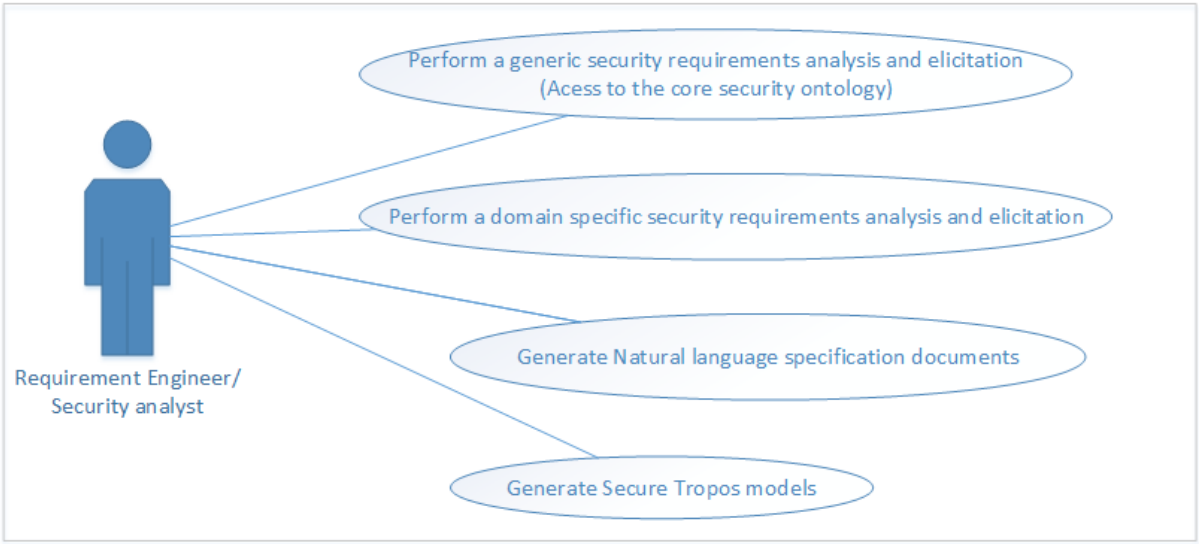


Figure D.1. Main functionalities of the tool.

Figure D.2 describes the technical architecture of the tool.

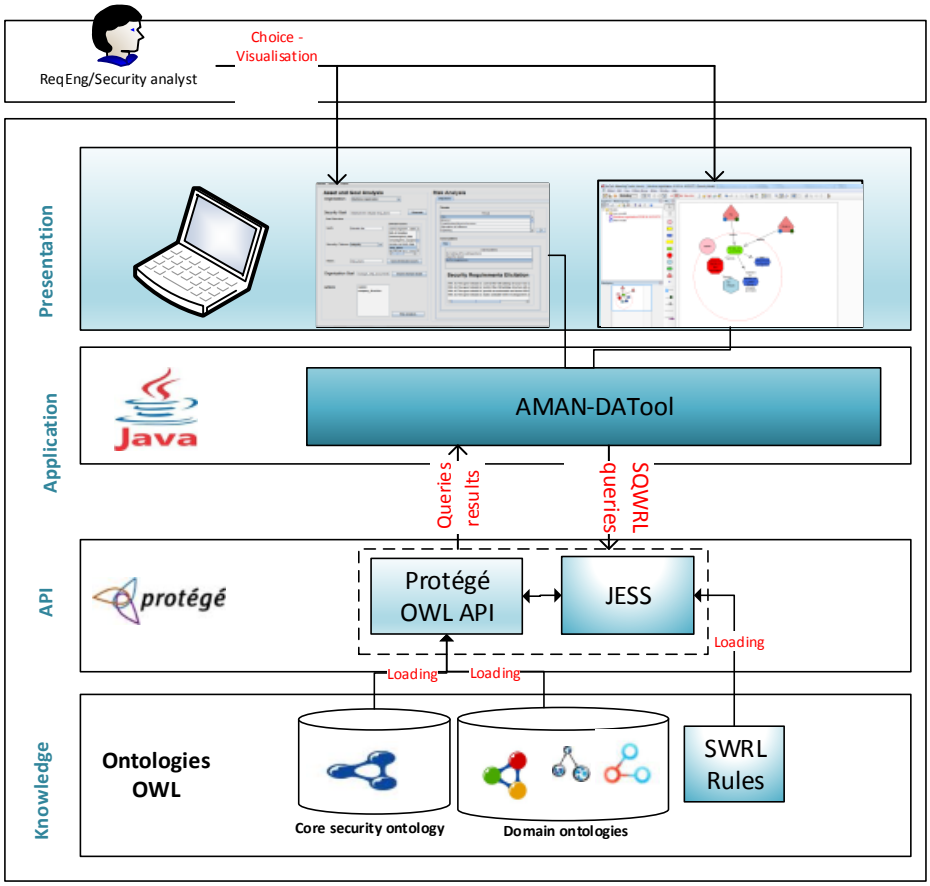


Figure D.2. Technical architecture of the tool

The technical architecture of the tool is organized around five main levels (user, presentation, application, API, knowledge). The user (requirements engineer or security analyst) use the tool AMAN-DATool through its presentation windows. He/she performs his/her security requirements analysis starting by introducing the security goals (choosing the assets to protect and the security criteria), analyzing the potential risk (choosing the potential threats, attackers, attack methods, and vulnerabilities); finally, generating the adequate domain specific security requirements. The different choices of the user are translated into SQWRL (Semantic Query-Enhanced Web Rule Language) queries that the tool generates dynamically and automatically. These queries are intended to the ontologies stored in the knowledge layer. The interaction between the presentation and application layers and the knowledge layer (ontologies) is ensured thanks to the APIs and the Jess engine[18]. At the end of the analysis, AMAN-DATool offers the generation of the specification document in addition to the Secure Tropos model that can be visualized with SecTro tool.

[18] http://www.jessrules.com/

Here follow some more details about each layer:

A. Knowledge

The core security ontology and the domain ontologies were edited using the Protégé (Horridge et al., 2004) editor, OWL language and SWRL rules.

There are different editors of ontologies that provide a variety of features and use different languages and formalisms, for instance, OntoEdit (Sure, Angele, and Staab, 2002), Ontolingua (Farquar, Fikes and Rice,1997). Our choice was to work with Protégé since it is an extensible, platform-independent environment for creating, editing, viewing, checking constraints, and extracting ontologies and knowledge bases. This choice was also motivated by the fact that Protégé has an intuitive and easy-to-use graphical user interface, is popular among the research community, and is also highly scalable.

Ontologies via Protégé can be developed in a variety of formats including OWL, RDF(S)[19], and XML Schema[20]. OWL (Web Ontology Language) was the language used for the development of the ontologies (security and domains ones). OWL is recommended by the World Wide Web Consortium (W3C). W3C states that OWL facilitates greater machine interpretability of Web content than XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics.

In addition to OWL-Protégé editing, SWRL (Semantic Web Rule Language) was used to edit the axioms (rules) of the ontologies. SWRL allows the definition of rules (Horrocks et al. , 2004) to enrich the semantic of OWL ontologies. The defined rules allow to deduct and to add new relations between the created individuals in an ontology. The structure of a SWRL rule respect the following schema:

```
antecedent ⇒ consequent
```

Where both *antecedent* and *consequent* are conjunctions of atoms (a_i) written $a_1 \wedge ... \wedge a_n$. An atom may refer to individuals, data literals, individual variables or data variables. Atoms can be of the form C(x), P(x,y), sameAs(x,y), differentFrom(x,y), where C is an OWL description or data range, P is an OWL property, x and y are either variables, OWL individuals or OWL data values, as appropriate. Variables are indicated using the standard convention of prefixing them with a question mark (e.g., ?x).

[19] http://www.w3.org/RDF/
[20] http://www.w3schools.com/schema/

Based on this syntax, rules (Rule1, Rule2) were defined. **Rule1** asserts that the composition of *Related* and *Satisfy* properties implies the *Enable* property. *Related* security goal (SG_x) to the security criterion (Cr_y) and *Satisfy* security requirement (SR_x) the security requirement (Cr_y) implies Enable (SR_z), (SG_x). As for **Rule 2**, vulnerability (Vul_x) that is MitigatedBy the security requirement (SR_y), threat (Th_z) that *Exploit* vulnerability (Vul_x) and threat (Th_z) that *Generates* ($Risk_t$) implies the *MitigatedBy* property between ($Risk_t$) and (SR_y).

Rule1. Related (?SG_x , ?Cr_y) ∧ Satisfy (?SR_z , ?Cr_y) ⇒ Enable (?SR_z , ?SGx)

Rule2. MitigatedBy (?Vul_x,?SR_y) ∧ Exploit (?Th_z, ?Vul_x) ∧ Generates (?Th_z, ?$Risk_t$) ⇒ MitigatedBy (?$Risk_t$, ?SR_y)

B. APIs

To be able to manipulate the ontologies through the AMAN-DATool Java application, the layer of APIs (Application Programming Interfaces) was introduced.

The Protégé-OWL[21] API is an open-source Java library for manipulating ontologies in OWL format. The API provides classes and methods to load and save OWL files, to query and manipulate OWL data models, and to perform reasoning based on SWRL rules. Furthermore, the API is optimized for the implementation of graphical user interfaces.

In our context, AMAN-DATool invokes this API to manipulate the security and domain ontologies.

Jess[22] (Java Expert System Shell) is an inference rules based engine completely developed in Java. It easily interfaces with applications developed in Java to add to them a reasoning layer. It has its own format of facts and rules but interprets also OWL ontologies, rules in SWRL format and SQWRL queries. Moreover, Jess interfaces with the Protégé ontology editor.

In our context, Jess was used to interpret and execute SQWRL queries on the security and domain ontologies during security requirements analysis and elicitation (asset and goal analysis, risk analysis and security requirements elicitation).

To test and extract relevant knowledge from the security and domain ontologies, SQWRL (Semantic Query-Enhanced Web Rule Language) was used. SQWRL is a SWRL-based for

[21] http://protegewiki.stanford.edu/wiki/ProtegeOWL_API_Programmers_Guide

querying OWL ontologies. It offers operators comparable to those of SQL (Structured Query Language) and exploits the semantics of SWRL rules for specifying the queries. SQWRL define a selection operator allowing the retrieval of instances described in an ontology. This operator is noted **`sqwrl:select.`** It also offers aggregation functions (such as **`max, min, avg, count`**, etc…), set operators and grouping functions. The detailed description of SQWRL syntax is beyond the scope of the study; readers may refer to O'Connor et al. (O'Connor and Das, 2009) for further details.

SQWRL queries are executed with the help of Jess rules engine. Some queries were already presented in chapter 5 (table 5.4). For example, the following query replies to the question "What are the vulnerabilities exploited by the threat Th_x?"

```
Exploits (Thx,?V)-> sqwrl:select(?V)
```

C. Application and presentation

The application and presentation layer of the AMAN-DATool has been developed using the Integrated Development Environment (IDE) Eclipse and the Java language. This layer interacts with the end user during the SRE process. Depending on end user choices, it displays the results of queries on the ontologies based on the other layers. It also generates the specification documents (a word document) that contain the security requirements. Moreover, it generates the Secure Tropos model in XML. These models can be visualized by the end user using the SecTro tool.

D. User

The tool is intended for requirements engineers who are asked to define security requirements within the requirements specification elaboration. It can also be used by security analysts for domain specific security analyses.

2 SECURITY REQUIREMENTS ANALYSIS AND ELICITATION USING AMAN-DATOOL

At the opening of the tool, the main window has a menu bar with the options (Security, Domain and Export) (see Figure D.3). The user can choose "Security" to perform a generic security requirements analysis and elicitation or "Domain" for a domain specific security

requirements analysis and elicitation. “Export” is intended to be used at the end of the analyses.

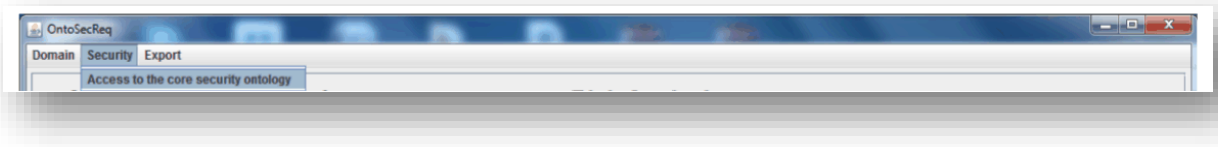


Figure D.3. The menu bar of the tool.

2.1. GENERIC SECURITY REQUIREMENTS ANALYSIS AND ELICITATION USING AMAN-DATOOL

If the user chooses “Security” in the menu bar, and clicks on “Access to the core security ontology”, a new window opens that contains the user interface allowing access to the core security ontology and performing a generic security requirements and elicitation. Figure D.4 shows a screenshot of that user interface.

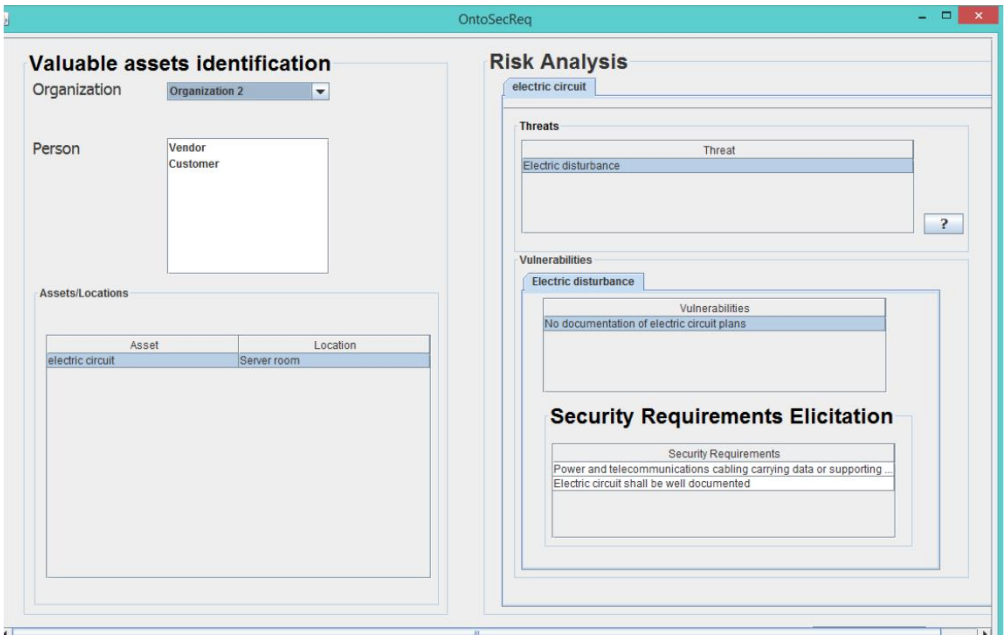


Figure D.4. Core Security ontology

The interface (Figure D.4) contains three parts corresponding to the three main steps of the SRE process: valuable assets identification (on the left side), risk analysis, and security requirements elicitation (on the right side).

This interface allows the user to choose the organization. It displays the persons involved and the list of all assets with their corresponding locations.

It also allows the user to choose valuable assets that he/she wants to protect. The latter are displayed at the left. For each asset the interface displays the corresponding threats (attackers and impact of each threat). For example, for the threat fire, a new window opens that suggest that this threat is led by a stowaway, and it has as impact the damage of organization assets (see Figure D.4 right-top). For each chosen threat, the interface displays the corresponding vulnerabilities. And finally, for all chosen vulnerabilities, the resulting list of security requirements to mitigate them is presented.

The "Save" button (right, down in Figure D.4) leads to the generation of the specification document that summarizes the analysis and the relevant security requirements.

2.2. DOMAIN SPECIFIC SECURITY REQUIREMENTS ANALYSIS AND ELICITATION USING AMAN-DATOOL

When the user clicks on "Domain" at the menu bar, he/she can access the different domains available in the knowledge layer, for which he/she can perform specific security requirements engineering process. The tool gives the possibility to add new domain ontologies dynamically so they can be available to the user (Figure D.5).

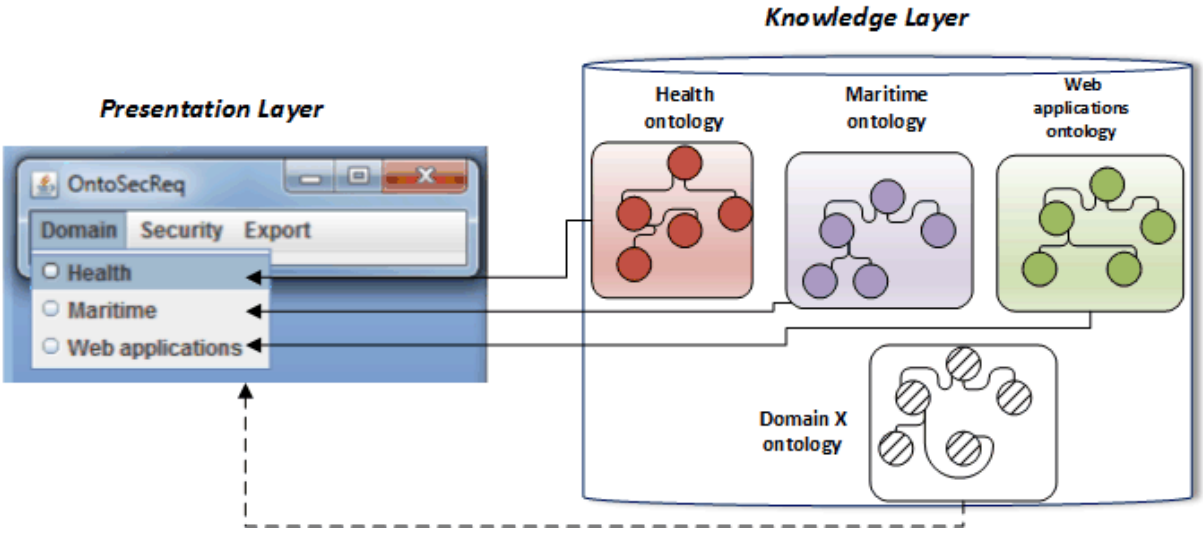


Figure D.5. Domain handling and selection.

Once a domain is chosen, the domain specific security requirements engineering process can start. It contains three main steps: goal and asset identification, risk analysis, security

requirements elicitation. It terminates in the generation of the SecTro model and the specification.

A. Goal and asset and analyses

The left part of the tool's main window (Figure D.6) is dedicated to the first step (asset and goal analysis).

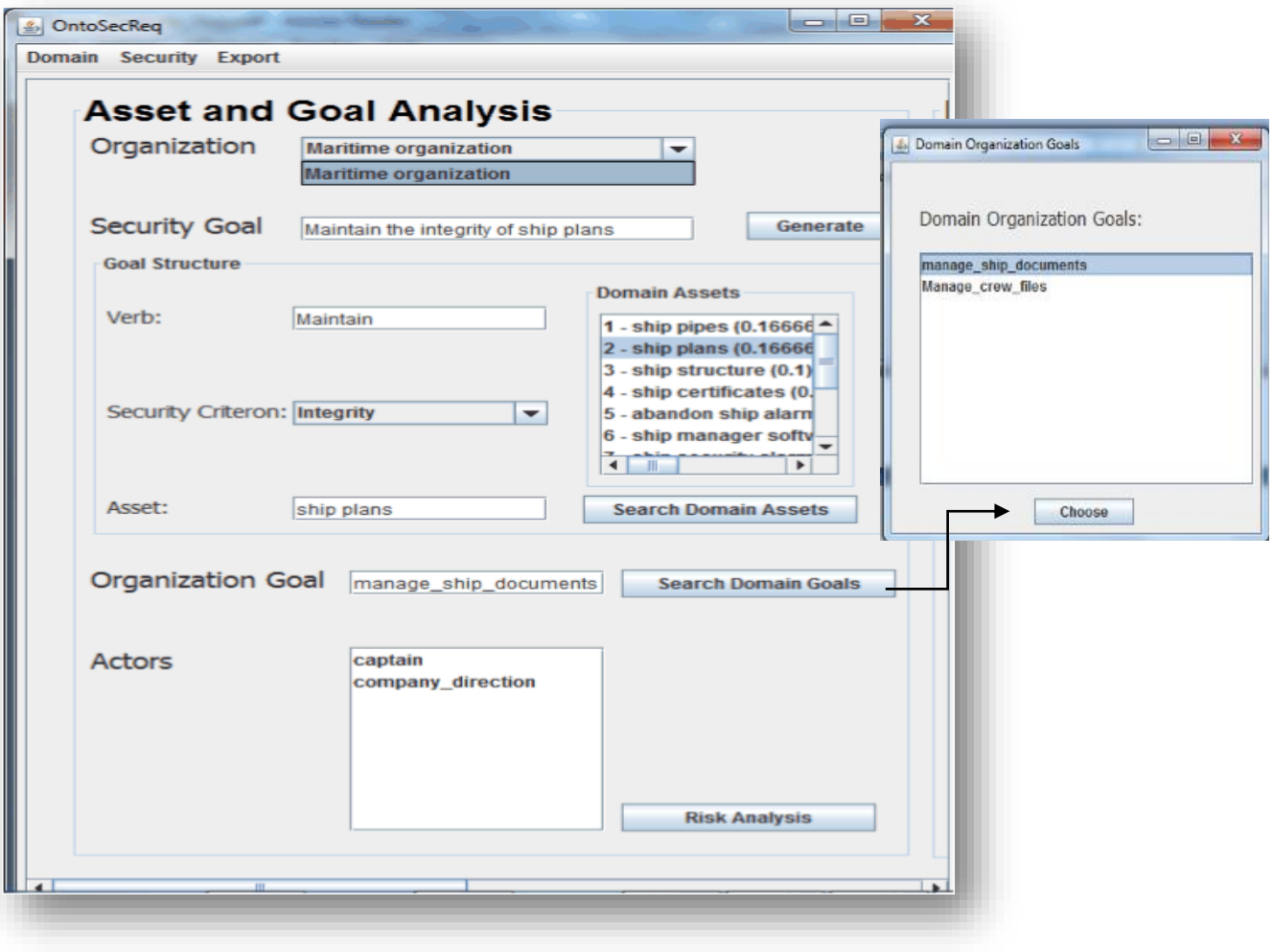


Figure D.6. Goal and asset analyzes

As Figure D.6 illustrates it, the tool allows the user to choose an organization of the domain (chosen before). Thus, each domain has different organizations in its scope.

Once the organization is selected, the user may form different security goals expressed by the stakeholder. The tool proposes a panel to formalize the security goal (according to the form presented in chapter 6) containing a textbox for its verb, selection box for the security criterion, and another textbox for the asset. For each asset introduced by the user, the tool

proposes its corresponding entry in the "Domain Assets" panel; this is to make sure that the introduced asset is a domain asset and not just any asset. When the security goal is formalized, the tool proposes the related organizational goals. Here for example: to the security goal "Maintain the integrity of ship plans"; corresponds the organizational goal: "manage the ship's documents". Finally, for each organizational goal chosen, the related actors are displayed to the user.

B. Risk analysis

The right-top part of the tool's main window is meant for the risk analysis (Figure D.7).

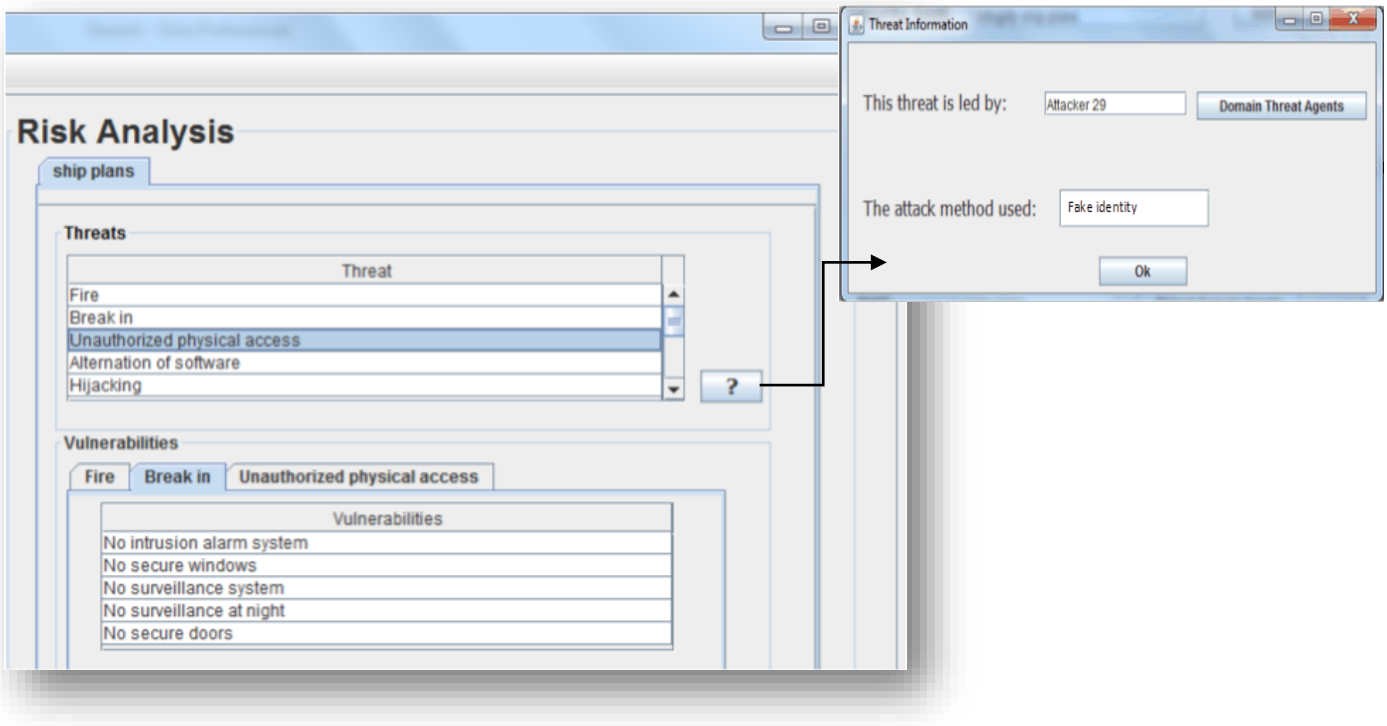


Figure D.7. Risk analysis

During the risk analysis step, the tool allows the user to select for each domain asset, the potential threats. For each threat selected, the user can have more information about the attacker who leads it and the attack method used. For each threat selected, the tool displays the list of potential exploited vulnerabilities.

C. Security requirements elicitation

The bottom-right part of the tool's main window (Figure D.8) is meant for the security requirements elicitation step.

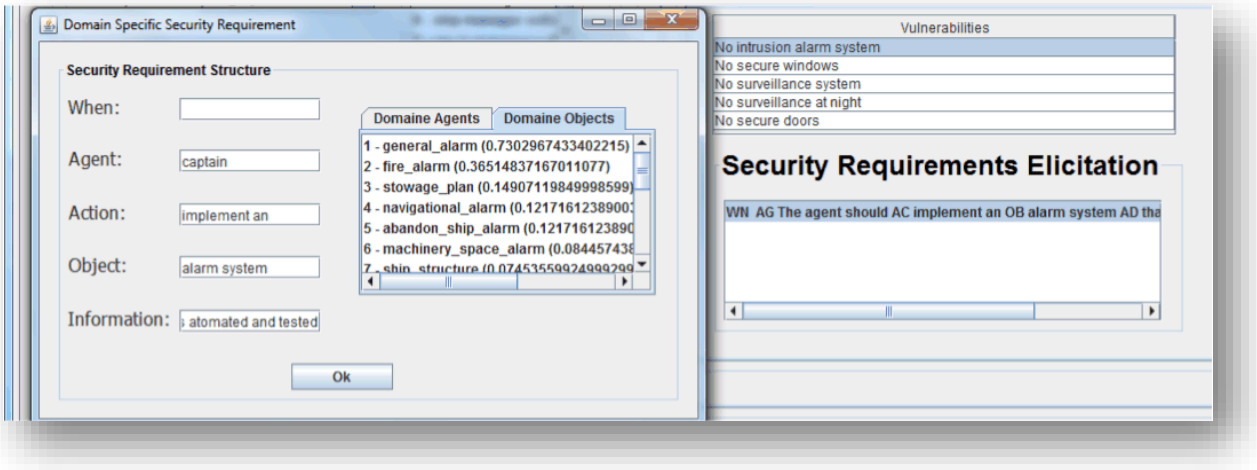


Figure D.8. Security requirements elicitation.

During the security requirements elicitation step, the tool provides (according to each selected vulnerability) the set of security requirements to consider. First these security requirements are displayed in their generic form. For example, as illustrated in the figure D.8, to the vulnerability (No intrusion alarm) corresponds the security requirement (WN AG The agent should AC implement an OB alarm system that is automated and tested).

Afterwards, the tool provides the user with the domain specific form of this security requirement. This is displayed in the window “Domain Specific Security Requirement”. The user can assign this requirement to another agent of the domain (AG). He/she can decide when to apply this requirement (WN), modify the action (AC) and choose another domain object (OB).

2.3. SECURITY REQUIREMENTS REPORT (SPECIFICATION) AND MODEL GENERATION

During the three previous steps, according to the user choices, the tool (Figure D.9) dynamically generates the specification and the SecTro model (it relies on the set of rules presented in chapter 7).

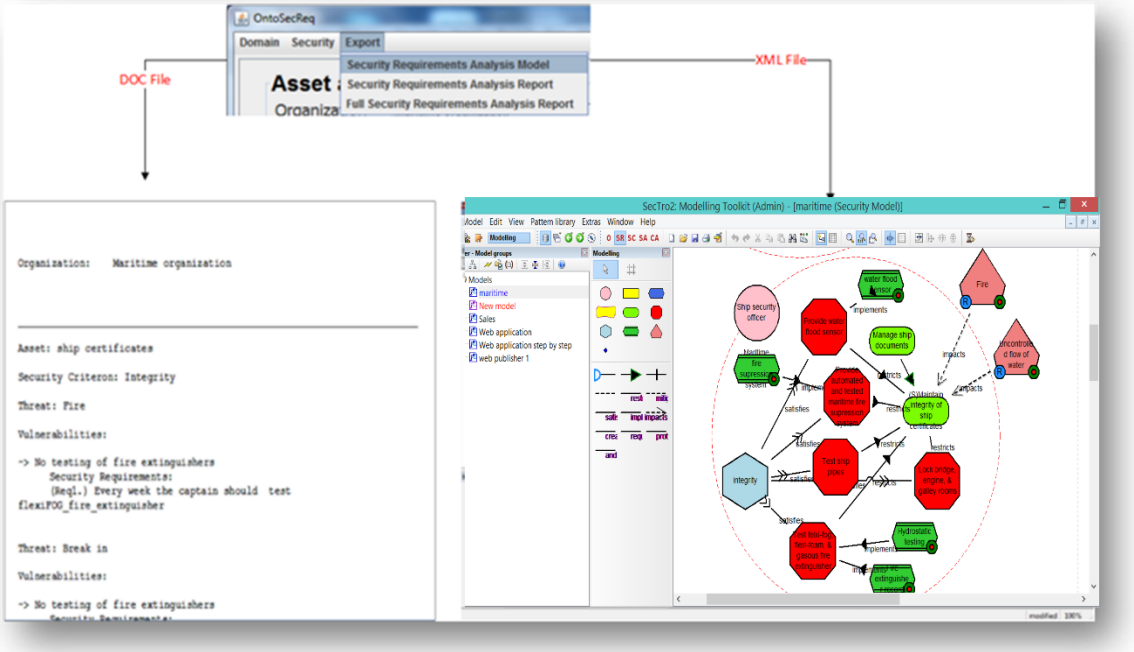


Figure D.9. Generation of security models or textual specifications

As Figure D.9 illustrates, the tool offers the user three choices of exportation:

- Security requirement analysis model: here, an XML file is created dynamically according to user selections in previous steps (and mapping and production rules presented earlier in chapter 8). This file can be displayed using the SecTro2 modeling toolkit. The user can then visualize his requirements models. With respect to Secure Tropos metamodel, the tool can generates the three views: organizational view, security requirement view and the attack view. Each view contains the adequate concepts and relations between them.
- Security requirement analysis report: here, a doc file is created dynamically. The document contains the information selected by the user. It includes the security requirements corresponding to the user's analysis.
- Full security requirements analysis report: this button allows the user to generate by one click the entire security requirements report – i.e., for a given security goal, all the possible threats, vulnerabilities and domain security requirements are generated.

3 CONCLUSION

In the future, the aim is to integrate AMAN-DATool with existing toolkits (SecTro2) for real time ontology based and model based security requirements analysis. We also try to improve the user interface to facilitate the job of the users.

Appendix E

Generated SecTro model in XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ADOXML SYSTEM "adoxml31.dtd">
<ADOXML version="3.1" date="26-12-14" time="13:56" database="adoxxdb" username="Admin"
adoversion="Version 1.0 4.0">
<MODELS>
<MODEL id="mod.11001" name="Maritime organization 26-12-14-13-56-39" version="1"
modeltype="Security Model" libtype="bp" applib="SecTro2_stable_git-ef66c55">
<MODELATTRIBUTES>
<ATTRIBUTE name="Keywords" type="STRING"/>
<ATTRIBUTE name="Description" type="STRING"/>
<ATTRIBUTE name="Comment" type="STRING"/>
<ATTRIBUTE name="Model type" type="ENUMERATION">Current model</ATTRIBUTE>
<ATTRIBUTE name="State" type="ENUMERATION">In process</ATTRIBUTE>
<ATTRIBUTE name="Reviewed on" type="STRING"/>
<ATTRIBUTE name="Reviewed by" type="STRING"/>
<ATTRIBUTE name="Author" type="STRING">Admin</ATTRIBUTE>
<ATTRIBUTE name="Creation date" type="STRING">14.01.2014, 09:38</ATTRIBUTE>
<ATTRIBUTE name="Last user" type="STRING">Admin</ATTRIBUTE>
<ATTRIBUTE name="Date last changed" type="STRING">26.02.2014,10:42:28</ATTRIBUTE>
<ATTRIBUTE name="Number of objects and relations" type="INTEGER">35</ATTRIBUTE>
<ATTRIBUTE name="Context of version" type="STRING"/>
<ATTRIBUTE name="Executive summary" type="LONGSTRING">EXEC SUMMARY: Executive summary not
provided.</ATTRIBUTE>
<ATTRIBUTE name="Organisational view" type="LONGSTRING">ORGANISATIONAL VIEW: Information not
provided.</ATTRIBUTE>
<ATTRIBUTE name="Security requirements view" type="LONGSTRING">SECURITY REQUIREMENTS VIEW:
Information not provided.</ATTRIBUTE>
<ATTRIBUTE name="Security components view" type="LONGSTRING">SECURITY COMPONENTS VIEW:
Information not provided.</ATTRIBUTE>
<ATTRIBUTE name="Security attacks view" type="LONGSTRING">SECURITY ATTACKS VIEW: Information
not provided.</ATTRIBUTE>
<ATTRIBUTE name="Cloud analysis view" type="LONGSTRING">CLOUD ANALYSIS VIEW: Information not
provided.</ATTRIBUTE>
<ATTRIBUTE name="Position" type="STRING"/>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="Demo settings attribute" type="STRING"/>
<ATTRIBUTE name="SecurityModelAttrrep" type="LONGSTRING">NOTEBOOK CHAPTER "Description" ATTR
"Keywords" ATTR "Description" ATTR "Comment" CHAPTER "User attributes" ATTR "Model type" ATTR
"State" ATTR "Reviewed on" ATTR "Reviewed by" CHAPTER "System attributes" ATTR "Author" ATTR
"Creation date" ATTR "Last user" ATTR "Date last changed" ATTR "Number of objects and
relations" ATTR "Context of version" CHAPTER "Executive summary" ATTR "Executive summary"
lines:20 CHAPTER "Organisational View" ATTR "Organisational view" lines:20 CHAPTER "Security
Requirements View" ATTR "Security requirements view" lines:20 CHAPTER "Security Components
View" ATTR "Security components view" lines:20 CHAPTER "Security Attacks View" ATTR "Security
attacks view" lines:20 CHAPTER "Cloud Analysis View" ATTR "Cloud analysis view"
lines:20</ATTRIBUTE>
<ATTRIBUTE name="_ModelBackgroundText" type="LONGSTRING">SRV - "Authentification
attack"</ATTRIBUTE>
<ATTRIBUTE name="SubviewGraphRep" type="LONGSTRING"> GRAPHREP layer:-1 AVAL
modelText:"_ModelBackgroundText" IF (LEN modelText) { FONT "Helvetica" h:14pt bold
color:darkgray TEXT (modelText) x:0.25cm y:0.25cm w:l h:t } </ATTRIBUTE>
</MODELATTRIBUTES>
<INSTANCE id="obj.1" class="Organisation" name="Organisation1">
<ATTRIBUTE name="Position" type="STRING">NODE x:2.5cm y:1.5cm w:6.5cm h:5.5cm index:3
visible:0 </ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="Show organisation name" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="Detailed description" type="LONGSTRING"/>
<ATTRIBUTE name="ChildObjects" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="_Master" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">Maritime organization</ATTRIBUTE>
```

<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING"> ATTR "Detailed description" ATTR "Show organisation name" ATTR "Position" </ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.2" class="Goal" name="Goal-2">
<ATTRIBUTE name="Position" type="STRING">NODE x:7cm y:6cm w:2cm h:1cm index:7 visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">manage_ship_documents</ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj"/>
<ATTRIBUTE name="Delegated to system" type="INTEGER">0</ATTRIBUTE>
</INSTANCE>
<CONNECTOR id="con.3" class="Is inside">
<FROM instance="Goal-2" class="Goal"></FROM>
<TO instance="Organisation1" class="Organisation"></TO>
<ATTRIBUTE name="AutoConnect" type="STRING"></ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.4" class="Goal" name="Goal-4">
<ATTRIBUTE name="Position" type="STRING">NODE x:7cm y:6cm w:2cm h:1cm index:7 visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">manage_ship_documents</ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-12-14-13-56-39" tmodelver="1" tclassname="Goal" tobjname="Goal-2"/>
</INTERREF>
<ATTRIBUTE name="Delegated to system" type="INTEGER">0</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.5" class="SecurityConstraint" name="SecurityConstraint-5">
<ATTRIBUTE name="Position" type="STRING">NODE x:5cm y:6.5cm w:2cm h:2cm index:13 visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">Maintain the confidentiality of ship plans</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
</INSTANCE>
<CONNECTOR id="con.6" class="Restricts">
<FROM instance="SecurityConstraint-5" class="SecurityConstraint"/>
<TO instance="Goal-4" class="Goal"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:14 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.7" class="SecurityObjective" name="SecurityObjective-7">
<ATTRIBUTE name="Position" type="STRING">NODE x:7cm y:9cm w:2cm h:2cm index:15 visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">Confidentiality</ATTRIBUTE>
</INSTANCE>
<CONNECTOR id="con.8" class="Satisfies">
<FROM instance="SecurityObjective-7" class="SecurityObjective"/>
<TO instance="SecurityConstraint-5" class="SecurityConstraint"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:16 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.9" class="Actor" name="Actor-9">
<ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:4.5cm w:2cm h:2cm index:5 visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">captain</ATTRIBUTE>
<ATTRIBUTE name="Show internal Goal Diagram" type="INTEGER">0</ATTRIBUTE>

```
<ATTRIBUTE name="_GoalDiagramHeight" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramWidth" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="Goal Diagram position" type="ENUMERATION">bottom-right</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING"> ATTR "Show internal Goal Diagram" ATTR
"Goal Diagram position" ATTR "Position" </ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj"/>
</INSTANCE>
<CONNECTOR id="con.10" class="Is inside">
<FROM instance="Actor-9" class="Actor"></FROM>
<TO instance="Organisation1" class="Organisation"></TO>
<ATTRIBUTE name="AutoConnect" type="STRING"></ATTRIBUTE>
</CONNECTOR>
<CONNECTOR id="con.11" class="DependencyLink">
<FROM instance="Goal-2" class="Goal"/>
<TO instance="Actor-9" class="Actor"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:9</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="Security Constraints" type="PROGRAMCALL">ITEM "Security Constraints list"
param:""</ATTRIBUTE>
<ATTRIBUTE name="_SecurityConstraints" type="LONGSTRING"/>
<ATTRIBUTE name="_LengthA" type="EXPRESSION">EXPR val:3</ATTRIBUTE>
<ATTRIBUTE name="_LengthB" type="EXPRESSION">EXPR val:1.5</ATTRIBUTE>
<ATTRIBUTE name="_endX" type="EXPRESSION">EXPR val:"4"</ATTRIBUTE>
<ATTRIBUTE name="_endY" type="EXPRESSION">EXPR val:"4.5"</ATTRIBUTE>
<ATTRIBUTE name="_NumberOfObjects" type="EXPRESSION">EXPR val:0</ATTRIBUTE>
<ATTRIBUTE name="_startX" type="EXPRESSION">EXPR val:"7"</ATTRIBUTE>
<ATTRIBUTE name="_startY" type="EXPRESSION">EXPR val:"6"</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING">ATTR "_SecurityConstraints"</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.12" class="Actor" name="Actor-12">
<ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:4.5cm w:2cm h:2cm index:5
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">company_direction</ATTRIBUTE>
<ATTRIBUTE name="Show internal Goal Diagram" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramHeight" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramWidth" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="Goal Diagram position" type="ENUMERATION">bottom-right</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING"> ATTR "Show internal Goal Diagram" ATTR
"Goal Diagram position" ATTR "Position" </ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj"/>
</INSTANCE>
<CONNECTOR id="con.13" class="Is inside">
<FROM instance="Actor-12" class="Actor"></FROM>
<TO instance="Organisation1" class="Organisation"></TO>
<ATTRIBUTE name="AutoConnect" type="STRING"></ATTRIBUTE>
</CONNECTOR>
<CONNECTOR id="con.14" class="DependencyLink">
<FROM instance="Actor-12" class="Actor"/>
<TO instance="Goal-2" class="Goal"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:9</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Organisational View</ATTRIBUTE>
<ATTRIBUTE name="Security Constraints" type="PROGRAMCALL">ITEM "Security Constraints list"
param:""</ATTRIBUTE>
<ATTRIBUTE name="_SecurityConstraints" type="LONGSTRING"/>
<ATTRIBUTE name="_LengthA" type="EXPRESSION">EXPR val:3</ATTRIBUTE>
<ATTRIBUTE name="_LengthB" type="EXPRESSION">EXPR val:1.5</ATTRIBUTE>
<ATTRIBUTE name="_endX" type="EXPRESSION">EXPR val:"4"</ATTRIBUTE>
<ATTRIBUTE name="_endY" type="EXPRESSION">EXPR val:"4.5"</ATTRIBUTE>
<ATTRIBUTE name="_NumberOfObjects" type="EXPRESSION">EXPR val:0</ATTRIBUTE>
<ATTRIBUTE name="_startX" type="EXPRESSION">EXPR val:"7"</ATTRIBUTE>
<ATTRIBUTE name="_startY" type="EXPRESSION">EXPR val:"6"</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING">ATTR "_SecurityConstraints"</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.15" class="Actor" name="Actor-15">
```

<ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:4.5cm w:2cm h:2cm index:5
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">captain</ATTRIBUTE>
<ATTRIBUTE name="Show internal Goal Diagram" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramHeight" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramWidth" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="Goal Diagram position" type="ENUMERATION">bottom-right</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING"> ATTR "Show internal Goal Diagram" ATTR
"Goal Diagram position" ATTR "Position" </ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Actor" tobjname="Actor-9"/>
</INTERREF>
<ATTRIBUTE name="_Master" type="INTEGER">1</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.18" class="AttackMethod" name="AttackMethod-18">
<ATTRIBUTE name="Position" type="STRING">NODE x:13.5cm y:7.5cm w:2.5cm h:1.25cm
index:8</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="NameST" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">null</ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"></IREF>
</INTERREF>
<INTERREF name="_ChildObj"></INTERREF>
</INSTANCE>
<INSTANCE id="obj.19" class="Attacker" name="Attacker-19">
<ATTRIBUTE name="Position" type="STRING">NODE x:10.5cm y:2.5cm w:10cm h:10cm
index:7</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="NameST" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">null</ATTRIBUTE>
<ATTRIBUTE name="Show internal Goal Diagram" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramHeight" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramWidth" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="Goal Diagram position" type="ENUMERATION">bottom-right</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"></IREF>
</INTERREF>
<INTERREF name="_ChildObj"></INTERREF>
</INSTANCE>
<INSTANCE id="obj.17" class="Threat" name="Threat-17">
<ATTRIBUTE name="Position" type="STRING">NODE x:15cm y:4.5cm w:2cm h:2cm index:7</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="NameST" type="STRING"></ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">Unauthorized physical access</ATTRIBUTE>
<ATTRIBUTE name="_HasSecurityAttacks" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_SecurityAttacksLink" type="PROGRAMCALL">ITEM "_SecurityAttacksLink"
param:"";</ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"></IREF>
</INTERREF>
<INTERREF name="_ChildObj"></INTERREF>
<ATTRIBUTE name="_InitialObjID" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_RebuildSAV" type="PROGRAMCALL">ITEM "_RebuildSAV" param:""</ATTRIBUTE>
</INSTANCE>
<CONNECTOR id="con.20" class="EmbodiesLink">
<FROM instance="Threat-17" class="Threat"/>
<TO instance="AttackMethod-18" class="AttackMethod"/>

```
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:23</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="_ThreatID" type="INTEGER">0</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.21" class="Goal" name="Goal-21">
<ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:5cm w:2cm h:1cm index:20</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">manage_ship_documents</ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Goal" tobjname="Goal-4"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"/>
</INTERREF>
<INTERREF name="_ChildObj"/>
<ATTRIBUTE name="Delegated to system" type="INTEGER">0</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.22" class="Vulnerability" name="Vulnerability-22">
<ATTRIBUTE name="Position" type="STRING">NODE x:4.5cm y:7cm w:2.6cm h:1.4cm
index:24</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">No entrance control</ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"/></INTERREF>
<INTERREF name="_ChildObj"/>
</INSTANCE>
<CONNECTOR id="con.23" class="Attacks">
<FROM instance="AttackMethod-18" class="AttackMethod"/>
<TO instance="Vulnerability-22" class="Vulnerability"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:25</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="_ThreatID" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_Mitigated" type="EXPRESSION">EXPR
expr:(tokcnt(cfobjs(ctobj(),"ProtectsLink"))) val:0</ATTRIBUTE>
</CONNECTOR>
<CONNECTOR id="con.24" class="AffectsLink">
<FROM instance="Vulnerability-22" class="Vulnerability"/>
<TO instance="Goal-21" class="Goal"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:21 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="_ThreatID" type="INTEGER">0</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.25" class=Constraint" name="Constraint-25">
<ATTRIBUTE name="Position" type="STRING">NODE x:10cm y:7cm w:2cm h:1cm index:27
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">controll the  ship_structure  access</ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj"/>
</INSTANCE>
<CONNECTOR id="con.26" class="Restricts">
<FROM instance="Goal-4" class="Goal"/>
<TO instance="Constraint-25" class="Constraint"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:29 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.27" class="Constraint" name="Constraint-27">
<ATTRIBUTE name="Position" type="STRING">NODE x:10cm y:7cm w:2cm h:1cm index:27
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">restrict the  ship_structure  access to only
local administrators</ATTRIBUTE>
```

<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj"/>
</INSTANCE>
<CONNECTOR id="con.28" class="Restricts">
<FROM instance="Constraint-27" class="Constraint "/>
<TO instance="Goal-4" class="Goal "/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:29 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
</CONNECTOR>
<INSTANCE id="obj.29" class="Actor" name="Actor-29">
<ATTRIBUTE name="Position" type="STRING">NODE x:4cm y:4.5cm w:2cm h:2cm index:5
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Attacks View@16</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">captain</ATTRIBUTE>
<ATTRIBUTE name="Show internal Goal Diagram" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramHeight" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="_GoalDiagramWidth" type="STRING">5cm</ATTRIBUTE>
<ATTRIBUTE name="Goal Diagram position" type="ENUMERATION">bottom-right</ATTRIBUTE>
<ATTRIBUTE name="Security criticality" type="INTEGER">0</ATTRIBUTE>
<ATTRIBUTE name="_SerialiseAttrs" type="LONGSTRING"> ATTR "Show internal Goal Diagram" ATTR
"Goal Diagram position" ATTR "Position" </ATTRIBUTE>
<INTERREF name="_ParentObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Actor" tobjname="Actor-15"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-16"/>
</INTERREF>
<INTERREF name="_ChildObj"/>
<ATTRIBUTE name="_Master" type="INTEGER">0</ATTRIBUTE>
</INSTANCE>
<INSTANCE id="obj.16" class="Threat" name="Threat-16">
<ATTRIBUTE name="Position" type="STRING">NODE x:6.5cm y:1.5cm w:2cm h:2.15cm index:17
visible:0</ATTRIBUTE>
<ATTRIBUTE name="External tool coupling" type="STRING"/>
<ATTRIBUTE name="NameST" type="STRING"/>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
<ATTRIBUTE name="Object's name" type="STRING">Unauthorized physical access</ATTRIBUTE>
<ATTRIBUTE name="_HasSecurityAttacks" type="INTEGER">1</ATTRIBUTE>
<ATTRIBUTE name="_SecurityAttacksLink" type="PROGRAMCALL">ITEM "_SecurityAttacksLink"
param:""</ATTRIBUTE>
<INTERREF name="_ParentObj"/>
<INTERREF name="_ChildObj">
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="AttackMethod" tobjname="AttackMethod-18"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Threat" tobjname="Threat-17"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Attacker" tobjname="Attacker-19"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Goal" tobjname="Goal-21 "/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Vulnerability" tobjname="Vulnerability-22"/>
<IREF type="objectreference" tmodeltype="Security Model" tmodelname="Maritime organization 26-
12-14-13-56-39" tmodelver="1" tclassname="Actor" tobjname="Actor-29"/>
</INTERREF>
<ATTRIBUTE name="_InitialObjID" type="INTEGER">16</ATTRIBUTE>
<ATTRIBUTE name="_RebuildSAV" type="PROGRAMCALL">ITEM "_RebuildSAV" param:""</ATTRIBUTE>
</INSTANCE>
<CONNECTOR id="con.30" class="ImpactsLink">
<FROM instance="Threat-16" class="Threat"/>
<TO instance="Goal-4" class="Goal"/>
<ATTRIBUTE name="Positions" type="STRING">EDGE 0 index:21 visible:0</ATTRIBUTE>
<ATTRIBUTE name="_View" type="STRING">Security Requirements View</ATTRIBUTE>
</CONNECTOR>
</MODEL>
</MODELS>
</ADOXML>

Appendix F

Evaluation form
(Controlled experiment)

Domain Specific Security Requirements Engineering

Aim of the experiment

Investigate the practices related to security requirements engineering during system development projects, identify the state of practice, and evaluate the proposed method and its tool- implementation against a controlled experiment with a case study and a questionnaire.

Participants

The experiment is mainly intended for people (industrials and academics) who have experiences in early phases of systems development (requirements elicitation and analysis phases) with a focus on security requirements elicitation.

Timing

You will be able to complete the experiment within 45 minutes**.**

Consent of participation and confidentiality

By filling in the questionnaire you consent to your voluntary participation in this experiment.

The data collected through the experiment will be kept confidential and will be stored securely, and will be deleted after completion of the experiment related activities. This questionnaire is anonym as well as the results obtained from the experiment.

We appreciate your time and effort! Thank you very much,

Amina Souag

Age: …
Sex: …
Position: …
Sector of activity: …

Phase 1: *pre-evaluation questions*

1. What is your relation to security? (Consultant, DSSI, PhD student, …)
2. How long has your experience with security been?
3. Do you refer to methods for security requirements elicitation? Which one(s)? Or do you rather use a ad-hoc technique (copy/paste for example)? Which one(s)?
4. Do these methods seem useful for you? Why?
5. Do these methods seem efficient for you? Why?
6. Which method (among the ones you cited) are the most efficient?
7. Do you know any security ontology before this experiment? If yes, which one(s)?
8. Do you want to know more about security ontologies?

9. Do you usually use a tool for the elicitation of security requirements? Which one(s) ?

Phase 2: *The Security ontology and its usability*

10. Do you find that the security ontology has the main concepts for security requirements elicitation?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:
11. Does the security ontology help in finding new elements (security requirements, threats, vulnerabilities, …)?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:
12. Does the security ontology help in building Secure Tropos models?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree

Comments:
13. Does the security ontology help in elicitating security requirements for specific domains? (The maritime domain for example)
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:

Phase 3: *Usage of the method I just presented to you*

14. Do you think the method makes an improvement in the elicitation of security requirements for specific domains?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:
15. Do you think that the method will be effective in discovering new security requirements for a specific domain?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree

Comments: *(Would you like to give any further comments and reply to the question compared to other methods)*
16. Do you think the method is overall useful?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:

Phase 4: *Usability of the method*

17. Do you think that the method reduces the effort required for the elicitation of security requirements?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:
18. Do you think that the method reduces the effort required to build security requirements models for different domains?
o Strongly agree o Agree o Neither agree nor disagree o Disagree o Strongly disagree
Comments:

19. Do you think the method is easy to use overall?
○ Strongly agree ○ Agree ○ Neither agree nor disagree ○ Disagree ○ Strongly disagree
Comments:
20. Do you have any further critics or modifications to suggest that improves the method?

Phase 5: *Usability of the prototype implementing the security ontology and the presented method*

21. Do you find the interface to access to the core security ontology easy to use?
○ Strongly agree ○ Agree ○ Neither agree nor disagree ○ Disagree ○ Strongly disagree
Comments:
22. Is this prototype friendly (clear and easy) to use?
○ Strongly agree ○ Agree ○ Neither agree nor disagree ○ Disagree ○ Strongly disagree

Comments:

23. Would you need help during the use of the prototype?

- Strongly agree
- Agree
- Neither agree nor disagree
- Disagree
- Strongly disagree

Comments:

24. Would you use the prototype in other projects? Which ones?

- Strongly agree
- Agree
- Neither agree nor disagree
- Disagree
- Strongly disagree

Comments:

Appendix G

Publications

This appendix contains the titles and venues of the publications the PhD research has produced so far:

Souag A., Mazo R., Salinesi C., Comyn-Wattiau I., Reusable knowledge in security requirements engineering: a systematic mapping study, Requirements Engineering Journal, 2015.

Souag A., Mazo R., Salinesi C., Comyn-Wattiau I. A Security Ontology for Security Requirements Elicitation, International Symposium on Engineering Secure Software and Systems, ESSoS 2015.

Souag A., Mazo R., Salinesi C., Comyn-Wattiau I. Using the AMAN-DA method to generate security requirements: a case study in the maritime domain, Requirements Engineering, Foundation for Software Quality, REFSQ 2016. Submitted.

Souag A. Bououd I. L'approche I*, Revue Genie logiciel, n°111, 2014.

Souag A., Salinesi C., Comyn-Wattiau I., A Methodology for Defining Security Requirements using Security and Domain Ontologies. Journal of International Council on Systems Engineering (Incosee). Insight, 2013.

Souag A., Salinesi C., Comyn-Wattiau I., Mouratidis H. Using Security and Domain ontologies for Security Requirements Analysis. Computer Software and Applications Conference Workshops (COMPSACW) 2013.

Souag A., Salinesi C., Comyn-Wattiau I., Security requirements analysis based on security and domain ontologies, REFSQ 2013.

Souag A. Une méthode de définition des exigences de sécurité fondée sur l'utilisation des ontologies. Séminaire Doctoral du Forum Académie-Industrie de l'AFIS, 2013.

Souag A., Salinesi C., Comyn-Wattiau I., Ontologies for security requirements: A literature survey and classification, Advanced Information Systems Engineering Workshops, 61-69, CAiSEW 2012.

Souag A., Towards a new generation of security requirements definition methodology using ontologies, 24th International Conference on Advanced Information Systems Engineering, CAiSE 2012.

Souag A., Vers une nouvelle génération de définition des exigences de sécurité fondée sur l'utilisation des ontologies, INFormatique des Organisation et des Systèmes d'Information de Décision, INFORSID 2012. **Best Paper Award.**

BIBLIOGRAPHY

Albrechtsen E. 2002. *A generic comparison of industrial safety and information security*. NTNU - Norwegian University of Science and Technology, http://www.iot.ntnu.no/users/albrecht/rapporter/generic%20comparision%20of%20ind%20saf%20and%20inf%20sec.pdf.

Alexander C., Ishikawa S., Silverstein M. 1977. *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press.

Alexander I. 2002. « Initial industrial experience of misuse cases in trade-off analysis ». In the proceedings *of the IEEE Joint International Conference on Requirements Engineering, 2002.*, 61-68.

Anderson, R. 2001. *Security engineering: A guide to building dependable distributed systems*. Wiley.

Anderson R. J., Anderson R. 2001. *Security Engineering: A Guide to Building Dependable Distributed Systems*. 1st ed. Wiley.

Antón A.I., Earp J.B. 2000. *Strategies for Developing Policies and Requirements for Secure Electronic Commerce Systems*. In E-commerce security and privacy, vol. 2, pp. 29-46.

Antón A.I., Earp J.B.. 2004. « A Requirements Taxonomy for Reducing Web Site Privacy Vulnerabilities ». *Requirements Engineering* 9 (3): 169-85.

Araujo R., Gupta S. 2005. « Design Authorization Systems Using SecureUML ». *Foundstone Professional Services*, 2-16.

Asnar, Y., Giorgini P., Massacci F., Zannone N.. 2007. « From Trust to Dependability through Risk Analysis ». In *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*, 19-26.

Asnar Y., Giorgini P., Mylopoulos J. 2006. *Risk Modelling and Reasoning in Goal Models*. Departmental Technical Report. University of Trento. http://eprints.biblio.unitn.it/959/.

Assali, A. A., Lenne D., Debray B. 2008. « Ontology development for industrial risk analysis ». In the *3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008.* 1-5. IEEE.

Avizienis A., Laprie J.-C., Randell B., Landwehr C. 2004. « Basic concepts and taxonomy of dependable and secure computing ». *IEEE Transactions on Dependable and Secure Computing*, 1 (1): 11-33.

Barnes S. 2005. « Assessing the value of IS journals ». *Communications of the ACM* 48 (1): 110-12.

Baskerville R. 1993. « Information systems security design methods: implications for information systems development ». *ACM Computing Surveys (CSUR)* 25 (4): 375-414.

Berry D. M. 1995. « The importance of ignorance in requirements engineering ». *Journal of Systems and Software* 28 (2): 179-84.

Best, B., Jurjens J., Nuseibeh B. 2007. « Model-Based Security Engineering of Distributed Information Systems Using UMLsec ». In *29th International Conference on Software Engineering, 2007. ICSE 2007*, 581-90.

Bittner T., Donnelly M., Barry Smith B. 2004. « Endurants and perdurants in directly depicting ontologies ». *AI Communications* 17 (4): 247-58.

Bjørner D. 2007. « A container line industry domain ». citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.127.5585

Bjørner D. 2010a. « Domain engineering ». In *Formal Methods: State of the Art and New Directions*, 1-41. Springer.

Bjørner D. 2010b. « Rôle of Domain Engineering in Software Development—Why Current Requirements Engineering Is Flawed ! ». In *Perspectives of Systems Informatics*, edited by Amir Pnueli, Irina Virbitskaite, Andrei Voronkov, 2-34. Lecture Notes in Computer Science 5947. Springer Berlin Heidelberg.

Blanco C., Velasco J., Fernández-Medina E., Valencia-García R., Toval A.. 2011. « Basis for an integrated security ontology according to a systematic review of existing proposals ». *Computer Standards & Interfaces* 33 (4): 372-88.

Blanco, C., Velasco J., Valencia-Garcia R., Fernandez-Medina E., Toval A., Piattini M. 2008. « A Systematic Review and Comparison of Security Ontologies ». In the *Third International Conference on Availability, Reliability and Security, 2008. ARES 08*, 813-20.

Borgol S., Carraraz M., Garloacz P., Vermaas P. 2009. « Towards the Ontologieal Representation of Functional Basis in DOLCE ».

Boyce S., Pahl C. 2007. *Developing Domain Ontologies for Course Content*. Educational Technology & Society-ETS, 10(3), 275-288.

Breaux T. D., Anton A.I. 2005. « Analyzing Goal Semantics for Rights, Permissions, and Obligations ». In *Proceedings of the 13th IEEE International Conference on Requirements Engineering*, 177-88. RE '05. Washington, DC, USA: IEEE Computer Society.

Bresciani P., Perini A., Giorgini P., Giunchiglia F., Mylopoulos J. 2004. « Tropos: An agent-oriented software development methodology ». *Autonomous Agents and Multi-Agent Systems* 8 (3): 203-36.

Buckshaw D. L., Parnell G.S., Unkenholz W.L., Parks D.L., Wallner J.M., Saydjari O.S. 2005. « Mission oriented risk and design analysis of critical information systems ». *Military Operations Research* 10 (2): 19-38.

Budgen D., Turner M., Brereton P., Barbara Kitchenham B. 2008. « Using mapping studies in software engineering ». Proceedings of PPIG Psychology of Programming Interest Group 2008, Lancaster University, UK (2008), pp. 195–204

Carvalho. J. A. 2000. « Information System? Which One Do You Mean? ». In *Proceedings of the IFIP TC8/WG8.1 International Conference on Information System Concepts: An Integrated Discipline Emerging*, 259-77. ISCO-4. Deventer, The Netherlands.

Castañeda V., Ballejos L., Caliusco M.L., Galli M.R.. 2010. « The Use of Ontologies in Requirements Engineering ». *Global Journal of Research Engineering* 10 (6).

Centre d'analyse stratégique. 2014. « Rapport annuel 2014 ».

Chandrasekaran B., Josephson J. R, and Benjamins V.R. 1999. "What are ontologies, and why do we need them?." IEEE Intelligent systems 1: 20-26.

Chebli A. S. 2009. « La piraterie maritime au début du XXième siecle: panorama, modes opératoires et solutions ». Mémoire pour le DU Analyse des menaces contemporaines.

Chernak, Y. 2012. « Requirements Reuse: The State of the Practice ». In *2012 IEEE International Conference on Software Science, Technology and Engineering (SWSTE)*, 46-53.

Chikh A., Abulaish M., Nabi S.I., Alghathbar K. 2011. « An Ontology Based Information Security Requirements Engineering Framework ». In *Secure and Trust Computing, Data Management and Applications*, edited by James J. Park, Javier Lopez, Sang-Soo Yeo, Taeshik Shon, David Taniar, 139-46. Communications in Computer and Information Science 186. Springer Berlin Heidelberg.

Christian T. 2010. *Security Requirements Reusability and the SQUARE Methodology*. No. CMU/SEI-2010-TN-027. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

CLUSIF. 2004. « Méthode Harmonisée d'Analyse de Risques (MEHARI), Principes et mécanismes ». http://www.clusif.asso.fr/.

CC, 2007. « Common Criteria for Information Technology Security Evaluation. Part 2: Security Functional components. ». Version 3.1. http://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R2.pdf

Dahl H. E. I., Hogganvik I., Stølen K. 2007. *Structured semantics for the CORAS security risk modelling language*. In the pre-proceedings of the 2nd International Workshop on Interoperability Solutions on Trust, Security, Policies and QoS for Enhanced Enterprise Systems (IS-TSPQ'07), pp. 79-92. 2007.

Daramola O., Sindre G., Moser T. 2012a. « Ontology-Based Support for Security Requirements Specification Process ». In *On the Move to Meaningful Internet Systems: OTM 2012 Workshops*, p 194-206. Lecture Notes in Computer Science 7567. Springer Berlin Heidelberg.

Daramola O., Sindre G., Stalhane T. 2012b. « Pattern-based security requirements specification using ontologies and boilerplates ». In *IEEE Second International Workshop on Requirements Patterns (RePa)*, p 54-59.

Davis F. D. 1989. « Perceived usefulness, perceived ease of use, and user acceptance of information technology ». *MIS quarterly*, 319-40.

Debenham J.K.1989. Knowledge systems design.. Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Denker G. 2002. « Access control and data integrity for daml+ oil and daml-s ». *SRI International, USA*.

Denker G., Kagal L., Finin T. 2005. « Security in the Semantic Web using OWL ». *Information Security, Technical Report* 10 (1): 51-58.

Denker G., Kagal L., Tim Finin, Paolucci M., Sycara K. 2003. « Security for DAML Web Services: Annotation and Matchmaking ». In *The Semantic Web Conferennce - ISWC 2003*, edited by Dieter Fensel, Katia Sycara, John Mylopoulos, 335-50. Lecture Notes in Computer Science 2870. Springer Berlin Heidelberg.

Denker G., Nguyen S., Ton A. 2004. « OWL-S Semantics of Security Web Services: a Case Study ». In *The Semantic Web: Research and Applications*edited by Christoph Bussler, John Davies, Dieter Fensel, Rudi Studer, 3053:240-53. Lecture Notes in Computer Science. Springer Berlin / Heidelberg.

Devanbu P. T., Stubblebine S. 2000. « Software engineering for security: a roadmap ». In *Proceedings of the Conference on the Future of Software Engineering*, 227-39. ACM.

Dik S. C. 1997. *The theory of functional grammar: the structure of the clause*. Walter de Gruyter.

Dobson G., Sawyer P. 2006. *Revisiting Ontology-Based Requirements Engineering in the age of the Semantic Web. In: Dependable Requirements Engineering of Computerised Systems at NPPs*.

Donner M. 2003. « Toward a Security Ontology ». *IEEE Security and Privacy* 1 (3): 6-7.

Dritsas, S., Gymnopoulos L., Karyda M., Balopoulos T., Kokolakis S., Lambrinoudakis C., Katsikas S. 2006. « A knowledge-based approach to security requirements for e-health applications ». *Electronic Journal for E-Commerce Tools and Applications*.

Dubois É., Heymans P., Mayer N., Matulevičius R. 2010. « A Systematic Approach to Define the Domain of Information System Security Risk Management ». In *Intentional Perspectives on Information Systems Engineering*, edited by Selmin Nurcan, Camille Salinesi, Carine Souveyet, Jolita Ralyté, 289-306. Springer Berlin Heidelberg.

Easterbrook S., Singer J., Storey M.A., Damian D. 2008. « Selecting empirical methods for software engineering research ». In *Guide to advanced empirical software engineering*, 285-311. Springer.

EBIOS., Secrétariat Général De la Défense Nationale, 2004. « EBIOS-Expression des Besoins et Identification des Objectifs de Sécurité ». http://www.ssi.gouv.fr/guide/ebios-2010-expression-des-besoins-et-identification-des-objectifs-de-securite/

ECMA. 1999. « Extended Commercially Oriented Functionality Class for Security Evaluation, E-COFC ». http://www.ecma-international.org/publications/files/ECMA-ST/ Ecma-271.pdf.

Ekelhart, A, Fenz S., Klemen M., Weippl E. 2007. « Security Ontologies: Improving Quantitative Risk Analysis ». In the *40th Annual Hawaii International Conference on,System Sciences HICSS.*

Elahi G., Yu E., Li T., Liu L. 2011. « Security Requirements Engineering in the Wild: A Survey of Common Practices ». In the IEEE 35th Annual Computer Software and Applications Conference (COMPSAC), 2011, pp. 314-319.

Elyan J. 2014. « 5 millions d'adresses et mots de passe Gmail postés sur un forum », septembre 11. http://www.lemondeinformatique.fr/actualites/lire-5-millions-d-adresses-et-mots-de-passe-gmail-postes-sur-un-forum-58602.html.

Evans, S., Heinbuch D., Kyle E., Piorkowski J., Wallner J. 2004. « Risk-based systems security engineering: stopping attacks with intention ». *IEEE Security Privacy* 2 (6): 59-62.

Fabian B., Gürses S., Heisel M., Santen T., Schmidt H. 2010. « A comparison of security requirements engineering methods ». *Requirements Engineering* 15 (1): 7-40.

Farquhar A. Fikes R., Rice J. 1997. « The Ontolingua Server: a tool for collaborative ontology construction ». *International Journal of Human-Computer Studies* 46 (6): 707-27.

Feigenbaum, E. A.; McCorduck, P. 1983, The fifth generation (1sted.), Reading, MA: Addison-Wesley, ISBN 978-0-201-11519-2, OCLC 9324691

Fenz S., Ekelhart A. 2009. « Formalizing information security knowledge ». In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, 183-94. ASIACCS '09. New York, NY, USA: ACM.

Fernández-López M., Gómez-Pérez A., Juristo N. 1997. « METHONTOLOGY: From Ontological Art Towards Ontological Engineering ». In *Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series*. Stanford University, EEUU: Facultad de Informática (UPM).

Fillmore C. J. 1967. « The case for case. »

Firesmith, D.G. 2007. « Engineering Safety and Security Related Requirements for Software Intensive Systems ». In *29th International Conference on Software Engineering - Companion, 2007. ICSE 2007 Companion*, p 169-169.

Firesmith D. G. 2004. « Specifying reusable security requirements ». *Journal of Object Technology* 3 (1): 61-75.

Firesmith D. G. 2003. « Security use cases ». *Journal of object technology* 2 (3). p 1-12.

Firesmith D. G. 2005. « A taxonomy of security-related requirements ». In *International Workshop on High Assurance Systems (RHAS'05)*.

Frakes, W.B., Kyo K. 2005. « Software reuse research: status and future ». *IEEE Transactions on Software Engineering* 31 (7): 529-36.

Frenkiel, Michel. 2009. « Cybercriminalité et crime organisé ». *ZDNet*. http://www.zdnet.fr/actualites/cybercriminalite-et-crime-organise-39386702.htm.

Geneiatakis D., Lambrinoudakis C. 2007. « An ontology description for SIP security flaws ». *Computer Communications* 30 (6): 1367-74.

Giorgini, Massacci F., Mylopoulos J., Zannone N. 2006. « Requirements Engineering for Trust Management: Model, Methodology, and Reasoning ». *International Journal of Information Security* 5 (4): 257-74

Giorgini, Massacci F., Mylopoulos J., Zannone N. 2005a. « Security and Trust Requirements Engineering ». In *Foundations of Security Analysis and Design III*, edited by Alessandro Aldini, Roberto Gorrieri, Fabio Martinelli, 237-72. Lecture Notes in Computer Science 3655. Springer Berlin Heidelberg.

Giorgini, Massacci F., Mylopoulos J., Zannone N. 2005b. « ST-tool: a CASE tool for security requirements engineering ». In *13th IEEE International Conference on Requirements Engineering, 2005. Proceedings*, 451-52.

Glinz, M. 2007. « On Non-Functional Requirements ». In the *15th IEEE International Requirements Engineering Conference, 2007. RE '07.*, 21-26.

Gruber T. R. 1995. « Toward principles for the design of ontologies used for knowledge sharing ». *International Journal of Human-Computer Studies* 43 (5–6): 907-28.

Grüninger M., Lee J. 2002. « Ontology Applications and Design-Introduction. » *Communication of the ACM* 45 (2): 39-41.

Gruselle B. 2013. « Enquête sur la sécurité numérique des entreprises ». Fondation pour la Recherche Stratégique (FRS), n°01/2013, https://www.frstrategie.org/barreFRS/publications/rd/2013/RD_201301.pdf

Gürses S. F., Berendt B., Santen T. 2006. « Multilateral security requirements analysis for preserving privacy in ubiquitous environments ». In *Proceedings of the UKDU Workshop*, 51-64.

Gürses S. F., Santen T. 2006. « Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation. » In *Sicherheit*, 6:42-53.

Guttman B., Aoback E.A. 1995. *An introduction to Computer Security: The NIST Handbook*. NIST. National Institute of Standards and Technology.

Haley, C.B., Laney R., Moffett J.D., and Nuseibeh B.. 2008. « Security Requirements Engineering: A Framework for Representation and Analysis ». *IEEE Transactions on Software Engineering* 34 (1): 133-53.

Haley, C. B., Moffett J.D, Laney R., and Nuseibeh B.. 2006. « A framework for security requirements engineering ». In *Proceedings of the 2006 international workshop on Software engineering for secure systems*, 35-42. SESS '06. New York, NY, USA: ACM.

Hatebur, D., Heisel M., Schmidt H. 2007. « A Pattern System for Security Requirements Engineering ». In *The Second RInternational Conference on Availability, Reliability and Security, 2007. ARES 2007*, 356-65.

He Q., Anton A.I. 2003. *A Framework for Modeling Privacy Requirements in Role Engineering*. In *Proceedings of Requirements Engineering Foundation for Software Quality, REFSQ*, vol. 3, pp. 137-146.

Hermoye L. A., Axel Van Lamsweerde A., Perry D. E. 2006a. « A Reuse-Based Approach to Security Requirements Engineering ». http://users.ece.utexas.edu/~perry/work/papers/060908-LH-reuse.pdf

Herrmann A., Morali A., Etalle S., Wieringa R. 2011. « RiskREP: risk-based security requirements elicitation and prioritization ». . In Perspectives in Business Informatics Research, in the 1st International Workshop on Alignment of Business Process and Security Modelling, 155 -162.

Herzog A., Shahmehri N., Duma D. 2007. « An Ontology of Information Security »: *International Journal of Information Security and Privacy* 1 (4): 1-23.

Hevner A., Chatterjee S. 2010. *Design Research in Information Systems: Theory and Practice*. 1st edition. Springer.

Hogganvik I., Stølen K. 2006. « A graphical approach to risk identification, motivated by empirical investigations ». In *Proceedings of the 9th international conference on Model Driven Engineering Languages and Systems*, 574-88. MoDELS'06. Berlin, Heidelberg: Springer-Verlag.

Horridge M., Knublauch H., Rector A., Stevens R., Wroe C. 2004. « A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0 ». *University of Manchester*.

Horrocks I, Patel-Schneider P.F., Boley H., Tabet S., Grosof B., Dean M. 2004. « SWRL: A semantic web rule language combining OWL and RuleML ». *W3C Member submission* 21: 79.

Hull E. 2011. *Requirements Engineering*. London: Springer-Verlag London Limited. http://dx.doi.org/10.1007/978-1-84996-405-0.

IMO. 1974. Inetrnational Maritime Organization. « International Convention for the Safety of Life at Sea (SOLAS), 1974 ». http://www.imo.org/About/Conventions/ListOfConventions/Pages/International-Convention-for-the-Safety-of-Life-at-Sea-%28SOLAS%29,-1974.aspx.

IMO 2011. International Maritim Organization. 2011. « ISPS Code ». http://www.imo.org/ourwork/security/instruments/pages/ispscode.aspx.

ISO-a. International Standard 15408-2, ISO/IEC. 1999. « ISO/IEC: Information Technology - Security Techniques - Evaluation Criteria for IT Security - Part 2: Security Functional Requirements ». Geneva Switzerland.

ISO-b. 2004.« ISO/IEC 13335-1:2004 Information technology -- Security techniques -- Management of information and communications technology security -- Part 1: Concepts and models for information and communications technology security management ». 2004.

ISO-c. 2009. ISO/IEC 15408. « Information technology -- Security techniques -- Evaluation criteria for IT security -- Part 1: Introduction and general model ».

Ivankina E. 2005. « An Approach to Guide Requirement Elicitation by Analysing the Causes and Consequences of Threats ». *Information Modelling and Knowledge Bases XVI* 121: 13.

Jackson M. J. 1995. *Software Requirements & Specifications: A Lexicon of Practice, Principles, and Prejudices*. New York; Wokingham, England; Reading, Mass.: ACM Press ; Addison-Wesley Pub. Co.

Jackson M. J. 2001. *Problem Frames: Analysing and Structuring Software Development Problems*. Harlow: Addison-Wesley/ACM Press.

Jacobson I., Booch G., Rumbaugh J. 1999. *The Unified Software Development Process*. Reading, Mass: Addison-Wesley.

Jensen J., Tøndel I.A, Meland H.P. 2010. « Experimental Threat Model Reuse with Misuse Case Diagrams ». In *Information and Communications Security*, editedby Miguel Soriano, Sihan Qing, Javier López, 355-66. Lecture Notes in Computer Science 6476. Springer Berlin Heidelberg.

Jones D., Bench-capon T., Visser P. 1998. « Methodologies For Ontology Development ».na , 62-75.

Jürjens J. 2002. « Using UMLsec and goal trees for secure systems development ». In *Proceedings of the 2002 ACM symposium on Applied computing*, 1026-30. SAC '02. New York, NY, USA: ACM.

Jürjens J. 2005. *Secure Systems Development with UML*. Berlin; New York: Springer.

Jürjens J., Shabalin P. 2004. « Automated verification of UMLsec models for security requirements ». In *UML 2004 – The Unified Modeling Language, volume 2460 of LNCS*, 412-25. Springer.

Kaiya, H., Saeki M. 2006. « Using Domain Ontology as Domain Knowledge for Requirements Elicitation ». In the *14th IEEE International Conference on Requirements Engineering*, ,189-98.

Karyda M., Balopoulos T., Dritsas S., Gymnopoulos L, Kokolakis S., Lambrinoudakis C., Gritzalis S. 2006. « An ontology for secure e-government applications ». In *the First International Conference on Availability, Reliability and Security, 2006. ARES 2006*, 5 pp.

Kim A., Luo J., Kang M. 2005. « Security Ontology for Annotating Resources ». In *Research Lab, NRL Memorandum Report*, 51.

Kitchenham, B.A., Pfleeger S.L., Pickard L.M., Jones P.W., Hoaglin D.C., El Emam,K.,Rosenberg J. 2002. « Preliminary guidelines for empirical research in software engineering ». *IEEE Transactions on Software Engineering* 28 (8): 721-34.

Kitchenham B.A., Budgen D., et Brereton O. P. 2011. « Using mapping studies as the basis for further research–a participant-observer case study ». *Information and Software Technology* 53 (6): 638-51.

Kitchenham B. A., Charters S. 2007. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Technical report, EBSE Technical Report EBSE-2007-01. http://www.dur.ac.uk/ebse/resources/Systematic-reviews-5-8.pdf.

Kotonya G., Sommerville I. 1998. *Requirements Engineering: Processes and Techniques*. Chichester; New York: J. Wiley.

Kusuma T.M. INTRODUCTION TO KNOWLEDGE ENGINEERING, Power point presentation, http://mkusuma.staff.gunadarma.ac.id/Downloads/files/29404/KBS-Review.pdf. Consulted in 25 June 2015.

Lam, W., McDermid J. A., Vickers A. J. 1997. « Ten Steps towards Systematic Requirements Reuse ». *Requirements Engineering* 2 (2): 102-13.

Landwehr C. E., Bull A.R., McDermott J.P., William S. C. 1994. « A taxonomy of computer program security flaws ». *ACM Computing Surveys (CSUR) 26, no. 3: 211-254.*

Lee Y., Lee J., Lee Z. 2002. « Integrating software lifecycle process standards with security engineering ». *Computers & Security* 21 (4): 345-55.

Lekhchine R. 2009. « Construction d'une ontologie pour le domaine de la sécurité : application aux agents mobiles ». Doctoral dissertation, PhD thesis, University Mentouri. Algeria

Lethbridge T. C. 2001. « Mixing Software Engineering Research and Development--What Needs Ethical Review and What Does Not? ». *Empirical Software Engineering* 6 (4): 319-21.

Levy Y., Ellis T. J. 2006. « A systems approach to conduct an effective literature review in support of information systems research ». *Informing Science: International Journal of an Emerging Transdiscipline* 9 (1): 181-212.

Lin L., Nuseibeh B., Ince D., Jackson M., Moffett J. 2003a. « Analysing security threats and vulnerabilities using abuse frames ». *Open University Technical Report No: 2003/10, 2003. Abuse Frame*

Lin L, Nuseibeh B., Ince D., Jackson M., Moffett J. 2003b. « Introducing abuse frames for analysing security requirements ». *In the proceedings of the 11th IEEE International Requirements Engineering Conference (RE'03)*, 371-72. IEEE Computer Society.

Lin L., Nuseibeh B., Ince D., Jackson M. "Using abuse frames to bound the scope of security problems." In the proceedings of the 12th IEEE International Requirements Engineering Conference, 2004, pp. 354-355.

Liu L., Yu E., Mylopoulos J. 2002. « Analyzing security requirements as relationships among strategic actors ». Submitted to the Symposium on Requirements Engineering for Information Security (SREIS'02), Raleigh, North Carolina. 2002.

Liu, L., Yu E., Mylopoulos J. 2003. « Security and privacy requirements analysis within a social setting ». In *Proceedings of the 11th IEEE International Requirements Engineering Conference, 2003* 151-61.

Lodderstedt T., Basin D., Doser J. 2002. « SecureUML: A UML-Based Modeling Language for Model-Driven Security ». In *≪UML≫ 2002 — The Unified Modeling Language*, edited by Jean-Marc Jézéquel, Heinrich Hussmann Stephen Cook, 426-41. Lecture Notes in Computer Science 2460. Springer Berlin Heidelberg.

Long, L. and Long, N. 1998. Computers, 5th ed. Prentice-Hall: Englewood Cliffs, NJ,
p. 5.

Longeon R., Archimbaud J. 1999. *Guide de la sécurité des systèmes d'information.* Centre National de la Recherche Scientifique.

López O., Laguna M. A., Peñalvo F.J.G. 2002. « Metamodeling for Requirements Reuse. » In WER02 - Workshop em Engenharia de Requisitos, Valencia, Spain (2002) *WER*, 76-90.

Lund M. S., Solhaug B., Stølen K. 2011. « The CORAS Tool ». In *Model-Driven Risk Analysis*, 339-46. Berlin, Heidelberg: Springer Berlin Heidelberg.

Maedche A., Staab S. 2001. « Ontology learning for the semantic web ». *IEEE Intelligent systems* 16 (2): 72-79.

Maiden N. 1993. *Integrated Software Reuse: Management and Techniques*. Aldershot, Hants, England ; Brookfield, Vt: Avebury Technical.

Martimiano, A. F. M., E. S. Moreira. 2005. « An owl-based security incident ontology ». In *Proceedings of the Eighth International Protege Conference*, 43-44.

Massacci F., Mylopoulos J., Paci F., Thein Thun Tun, Yu Y. 2011. « An extended ontology for security requirements ». In *Advanced Information Systems Engineering Workshops*, 622-36. Springer.

Massacci F., Mylopoulos J., Zannone N. 2007. *An Ontology for Secure Socio-Technical Systems*. Handbook of ontologies for business interaction 1 (2007): 469.

Massacci F., Prest M., Zannone N. 2004. *Using a Security Requirements Engineering Methodology in Practice: the compliance with the Italian Data Protection Legislation*. Departmental Technical Report. University of Trento. http://eprints.biblio.unitn.it/702/.

Massacci F., Zannone N. 2008. « Detecting Conflicts between Functional and Security Requirements with Secure Tropos: John Rusnak and the Allied Irish Bank ». *Social modeling for requirements engineering. MIT Press, Cambridge*.

Mayer N. 2012. *Model-Based Management of Information System Security Risk*. Presses universitaires de Namur.

Mayer N., Rifaut A., Dubois E. 2005. « Towards a risk-based security requirements engineering framework ». *In Proceedings of the Workshop on Requirements Engineering for Software Quality REFSQ*. Vol. 5

McDermott J., Fox C. 1999. « Using abuse case models for security requirements analysis ». in the *Proceedings* of the 15th annual *Computer Security Applications Conference, 1999.(ACSAC'99)*.55-64.

MD, NATIONAL SECURITY AGENCY/CENTRAL SECURITY SERVICE FORT GEORGE G. MEADE. 2002. *Common Criteria for Information Technology Security Evaluation: Department of Defense Public Key Infrastructure and Key Management Infrastructure Token Protection Profile (Medium Robustness)*.

Mead N. R., Stehney T. 2005. « Security quality requirements engineering (SQUARE) methodology ». In *Proceedings of the 2005 workshop on Software engineering for secure system & building trustworthy applications*, 1-7. SESS '05. New York, NY, USA: ACM.

Mead N. R., Viswanathan V., Padmanabhan D., Raveendran A. 2008. *Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models*. Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models. Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models.

Mead N.R., Hough E.D. 2006. « Security Requirements Engineering for Software Systems: Case Studies in Support of Software Engineering Education ». In *Proceedings of the 19th Conference on Software Engineering Education and Training, 2006.*, 149-58.

Meier, J. D. 2006. « Web application security engineering ». *Security & Privacy, IEEE* 4 (4): 16-24.

Mellado D., Blanco C., Sánchez L.E., Fernández-Medina E. 2010. « A systematic review of security requirements engineering ». *Computer Standards & Interfaces* 32 (4): 153-65.

Mellado, D., Fernandez-Medina E., Piattini M., 2006. « Applying a Security Requirements Engineering Process ». In *Computer Security – ESORICS 2006*, edited by Dieter Gollmann, Jan Meier, Andrei Sabelfeld, 192-206. Lecture Notes in Computer Science 4189. Springer Berlin Heidelberg.

Mellado, D., Fernandez-Medina E., Piattini M. 2007. « A common criteria based security requirements engineering process for the development of secure information systems ». *Computer standards & interfaces* 29 (2): 244-53.

Mellado, D., Fernandez-Medina E., Piattini M. 2008. « Security Requirements Variability for Software Product Lines ». In *Third International Conference on Availability, Reliability and Security, 2008. ARES 08*, 1413-20.

Metcalfe, M. 2002. « Metaphors for literature reviews (responses summary for ISWorld listserv email request) ».

Miller G.A. 1995. « WordNet: A Lexical Database for English ». *Commun. ACM* 38 (11): 39-41.

Mohan R., Arumugam G. 2005. « Constructing Railway Ontology using Web Ontology Language and Semantic Web Rule Language ». *International Journal of Computer Technology and Applications*.

Moody D. 2003. « The Method Evaluation Model: A Theoretical Model for Validating Information Systems Design Methods ». *in Proceedings of the 11th European Conference on Information Systems , C. U. Ciborra, R. Mercurio, M. DeMarco, M. Martinez, and A. Carignani (eds.), Naples, Italy, June 16-21*

Morali, A., Wieringa R. 2010. « Risk-based Confidentiality Requirements Specification for Outsourced IT Systems ». In *18th IEEE International Requirements Engineering Conference (RE), 2010*, 199-208.

Morali, A., Zambon, E., Etalle, S. & Wieringa, R.J. (2009). CRAC : Confidentiality risk analysis and IT-architecture comparison of business networks. (External Report, CTIT Technical Report, No. TR-CTIT-09-30). Enschede: Universiteit Twente, 8 pp

Mouratidis H. 2006. « Analysing Security Requirements of Information Systems using Tropos ». roceedings 1st Annual Conference on Advances in Computing and Technology (AC&T), London - United Kingdom, pp. 55 – 64.

Mouratidis H., Giorgini P. 2007. « *Secure Tropos: A Security-Oriented Extension of the Tropos Methodology*.». *International Journal of Software Engineering and Knowledge Engineering* 17 (02): 285-309.

Mouratidis H., Giorgini P., and Manson G.. 2003a. « Integrating security and systems engineering: Towards the modelling of secure information systems ». In *In Proceedings of the 15th Conference On Advanced Information Systems Engineering (CAiSE*, 63-78. Springer-Verlag.

Mouratidis H., Giorgini P., and Manson G.. 2003b. « An ontology for modelling security: The tropos approach ». In *Knowledge-Based Intelligent Information and Engineering Systems*, 1387-94. Springer.

Mouratidis H., Giorgini P., and Manson G. 2005. « When security meets software engineering: a case of modelling secure information systems ». *Information Systems* 30 (8): 609-29.

Mouratidis H., Giorgini P., Schumacher M., Manson G. 2003. « Security patterns for agent systems ». In *Proceedings of the Eight European Conference on Pattern Languages of Programs (EuroPLoP), Irsee, Germany*.

Mouratidis H., Islam S., Kalloniatis C., Gritzalis S. 2013. « A framework to support selection of cloud providers based on security and privacy requirements ». *Journal of Systems and Software* 86 (9): 2276-93.

Mouratidis H., Weiss M., Giorgini P. 2006. « Modelling Secure Systems Using an Agent-Oriented Approach and Security Patterns ». *International Journal on Software Engineering and Knowledge Engineering.* 16: 471-98.

Mylopoulos J., Borgida A., Jarke M., Koubarakis M. 1990. « Telos: Representing knowledge about information systems ». *ACM Transactions on Information Systems (TOIS)* 8 (4): 325-62.

Najar S., Kirsch Pinheiro M., Souveyet C., Steffenel L.A. 2012. « Service discovery mechanism for an intentional pervasive information system ». In *19th International Conference on Web Services (ICWS), 2012 IEEE*, 520-27. IEEE.

Newell A. 1982. The knowledge level. Artificial Intelligence, 18:87–127.

Nguyen V., 2011. « Ontologies and information systems: a literature survey ». Defence Science and Technology Organisation, Technical Note DSTO-TN-1002.

Norton. 2013. « 2013 Norton Report ».

O'Connor M. J., Das. A. K , 2009. « SQWRL: A Query Language for OWL. » In *OWLED*. Vol. 529.

Okubo, T., Kaiya H., Yoshioka N. 2011. « Effective Security Impact Analysis with Patterns for Software Enhancement ». In *Sixth International Conference on Availability, Reliability and Security (ARES)*, 527-34.

MAGERIT, 2012. « PAe - MAGERIT v.3 : Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información ». http://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Magerit.html#.UhAEVLxsDR0.

Paja E., Dalpiaz F., Poggianella M., Roberti P., Giorgini P. 2012. « STS-Tool: Using Commitments to Specify Socio-Technical Security Requirements ». In *Advances in Conceptual Modeling*, edited by Silvana Castano, Panos Vassiliadis, Laks V. Lakshmanan, Mong Li Lee, 396-99. Lecture Notes in Computer Science 7518. Springer Berlin Heidelberg.

Paolucci M., Kawamura T., Payne T.R., Sycara K. 2002. « Semantic Matching of Web Services Capabilities ». In *The Semantic Web — ISWC 2002*, edited by Ian Horrocks, James Hendler, 333-47. Lecture Notes in Computer Science 2342. Springer Berlin Heidelberg.

Pavlidis M., Mouratidis H., Kalloniatis C., Shareeful I., Gritzalis S. 2013. « Trustworthy Selection of Cloud Providers Based on Security and Privacy Requirements: Justifying Trust Assumptions ». In *Trust, Privacy, and Security in Digital Business*, edited by Steven Furnell, Costas Lambrinoudakis, Javier Lopez, 185-98. Lecture Notes in Computer Science 8058. Springer Berlin Heidelberg.

Pearsall J., Patrick H., ed. 2001. *New Oxford Dictionary of English*. New York: Oxford University Press.

Peffers K., Tuunanen T., Rothenberger M. A., Chatterjee S. 2007. « A design science research methodology for information systems research ». *Journal of management information systems* 24 (3): 45-77.

Petersen K., Feldt R.,Mujtaba S., Mattsson M. 2008. « Systematic mapping studies in software engineering ». In *12th International Conference on Evaluation and Assessment in Software Engineering*, 17:1.

Pfleeger, Shari Lawrence. 1995. « Experimental design and analysis in software engineering ». *Annals of Software Engineering* 1 (1): 219-53.

Pohl K. 2010. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer.

Potts C. 1997. « Fitness for use: the system quality that matters most ». In the *proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ*. Vol. 97. , 97:15-28.

Prat N. 1997. « Goal formalisation and classification for requirements engineering ». *Proceedings of Requirements Engineering: Foundation for Software Quality*.

Rainer Jr., Kelly R., Miller M.D. 2005. « Examining differences across journal rankings ». *Communications of the ACM* 48 (2): 91-94.

Rannenberg K. 1993. « Recent Development in Information Technology Security Evaluation-The Need for Evaluation Criteria for Multilateral Security. » In *Security and Control of Information Technology in Society*, 113-28.

Robertson S., and Robertson J.. 2013. *Mastering the Requirements Process Getting Requirements Right*. Upper Saddle River, N.J.: Addison-Wesley.

Robson C. 2011. *Real world research: a resource for users of social research methods in applied settings*. Wiley Chichester.

Rolland C., Salinesi C. 2005. « Modeling goals and reasoning with them ». In *Engineering and Managing Software Requirements*, 189-217. Springer.

Rolland, C., Souveyet C, Benachour C. 1998. « Guiding goal modeling using scenarios ». *IEEE Transactions on Software Engineering* 24 (12): 1055-71.

Runeson P., Host M., Rainer A., Regnell B.. 2012. *Case Study Research in Software Engineering: Guidelines and Examples*. 1 edition. Hoboken, N.J: Wiley.

Rupp C., Simon M., Hocker F. 2009. « Requirements engineering und management ». *HMD Praxis der Wirtschaftsinformatik* 46 (3): 94-103.

Rushby J. 2001. « Security requirements specifications: How and what ». In *Symposium on Requirements Engineering for Information Security (SREIS)*. Vol. 441.

Saeki M., Kaiya H. 2009. « Security Requirements Elicitation Using Method Weaving and Common Criteria ». In *Models in Software Engineering*, edited by Michel R. V. Chaudron, 185-96. Lecture Notes in Computer Science 5421. Springer Berlin Heidelberg.

Salinesi C., Ivankina E., Angole W. 2008. « Using the RITA Threats Ontology to Guide Requirements Elicitation: an Empirical Experiment in the Banking Sector ». In the *First International Workshop on Managing Requirements Knowledge, 2008. MARK '08.*, 11-15.

Salini, P., Kanmani S. 2012a. « A Knowledge-oriented Approach to Security Requirements for an E-Voting System ». *International Journal of Computer Applications* 49 (11): 21-25.

Salini, P., Kanmani S. 2012b. « Survey and analysis on Security Requirements Engineering ». *Computers & Electrical Engineering* 38 (6): 1785-97.

Schneier B. 2008. « Beyond fear ». *Copernicus Book, New York*.

Semmak, F., Gnaho C., Laleau R. s. d. « Extended Kaos to Support Variability for Goal Oriented Requirements Reuse ». In Proceedings of the international workshop on model driven information systems engineering: enterprise, user and system models, CEUR, pp 22–33

Sindre, G., Opdahl A. L. 2000. « Eliciting security requirements by misuse cases ». In the *37th International Conference on Technology of Object-Oriented Languages and Systems, 2000. TOOLS-Pacific 2000. Proceedings*, 120-31.

Sindre G., Firesmith D.G., Opdahl A.L. 2003. « A Reuse-Based Approach to Determining Security Requirements ». In *Proceedings of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03)*, 16-17.

Sindre G., Opdahl A.L. 2001a. « Capturing security requirements through misuse cases ». *NIK 2001, Norsk Informatik konferanse 2001*.

Sindre G., Opdahl A.L. 2001b. « Templates for Misuse Case Description ». In *Proceedings of the 7th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'2001)*, 4-5.

Sindre G., Opdahl A.L. 2005. « Eliciting Security Requirements with Misuse Cases ». *Requirements engineering* 10, no. 1 (2005): 34-44.

Souag, A. 2012. « Towards a new generation of security requirements definition methodology using ontologies ». In *24th International Conference on Advanced Information Systems Engineering (CAiSE'12) Gdańsk, Poland, 25 - 29 June 2012*, 1-8.

Souag A., Salinesi C., Comyn-Wattiau I. 2012. « Ontologies for Security Requirements: A Literature Survey and Classification ». In *Advanced Information Systems Engineering Workshops*, edited by Marko Bajec and Johann Eder, 112:61-69. Berlin, Heidelberg: Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007%2F978-3-642-31069-0_5.

Staab S., Maedche A. 2000. *Axioms are Objects, too – Ontology Engineering beyond the Modeling of Concepts and Relations*. Proceedings of the ECAI 2000 Workshop on Ontologies and Problem - Solving Methods, Berlin, Germany.

Sure Y., Angele J., Staab S. 2002. « OntoEdit: Guiding Ontology Development by Methodology and Inferencing ». In *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and*

ODBASE, edited by Robert Meersma Zahir Tari, 1205-22. Lecture Notes in Computer Science 2519. Springer Berlin Heidelberg.

Susi A, Perini A., Mylopoulos J., Giorgini P. 2005. « The tropos metamodel and its use ». *Informatica (Slovenia)* 29 (4): 401-8.

Symantec. 2013. « Symantec 2013 Annual Report ». http://investor.symantec.com/files/doc_financials/Annual%20Reports/Symantec%202013%20Annual%20Report.pdf.

Tettero, O., Out D. J., Franken H. M, Schot J. 1997. « Information security embedded in the design of telematics systems ». *Computers & Security* 16 (2): 145-64.

Tondel I. A., Jaatun M. G., and Meland P. H.. 2008. « Security requirements for the rest of us: A survey ». *Software, IEEE* 25 (1): 20-27.

Toninelli A. Corradi A., Montanari R. 2008. « Semantic-based discovery to support mobile context-aware service access ». *Computer Communications* 31 (5): 935-49.

Toval A., Nicolás J., Moros B., García O. 2001. « Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach ». *Requirements Engineering Journal* 6: 205-19.

Tsoumas, B., Gritzalis D. 2006. « Towards an Ontology-based Security Management ». In the *20th International Conference on Advanced Information Networking and Applications, 2006. AINA 2006*, 1:985-92.

Undercoffer J., Joshi A., Pinkston J. 2003. « Modeling Computer Attacks: An Ontology for Intrusion Detection ». In *the 6th International Symposium on Recent Advances in Intrusion Detection*, 113-35. Springer.

Uschold M., Gruninger M. 1996. « Ontologies: Principles, methods and applications ». *Knowledge Engineering Review* 11: 93-136.

Van Lamsweerde A. 2001. « Goal-oriented requirements engineering: a guided tour ». In the *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering, 2001.*, 249-62.

Van Lamsweerde A. 2004. « Elaborating security requirements by construction of intentional anti-models ». In the *proceedings of the 26th International Conference on Software Engineering, 2004. ICSE 2004.*, 148-57.

Van Lamsweerde A. 2007. « Engineering Requirements for System Reliability and Security ». Nato Security Through Science Series D-Information and Communication Security 9 (2007): 196.

Van Lamsweerde, A. 2009. « From worlds to machines ». *A Tribute to Michael Jackson*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.156.2270&rep=rep1&type=pdf

Velasco J. L., Valencia-Garcia R., Fernandez-Breis J. T., Toval A. 2009. « Modelling Reusable Security Requirements Based on an Ontology Framework ». *Journal of Research and Practice in Information Technology* 41 (2): 119.

Viljanen L. 2005. « Towards an ontology of trust ». In *Trust, Privacy, and Security in Digital Business*, 175-84. Springer.

Vogel V. 2013. « Information Security Guide ». consulted 6 december 2013, https://wiki.internet2.edu/confluence/display/itsg2/Overview+to +the+Guide.

Vorobiev A., Han J. 2006. « Security attack ontology for web services ». In the *Second International Conference on Semantics, Knowledge and Grid, 2006. SKG'06.*, 42-42. IEEE.

Vraalsen F., den Braber F., Lund M.S., Stølen K. 2005. « The CORAS Tool for Security Risk Analysis ». In *Trust Management*, edited by Peter Herrmann, Valérie Issarny, Simon Shiu, 402-5. Lecture Notes in Computer Science 3477. Springer Berlin Heidelberg.

Wang X. H., Zhang D.Q., Gu T., and Pung H.K. "Ontology based context modeling and reasoning using OWL." In Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004 pp. 18-22.

Wenzel S., Warzecha D., Jurjens J. 2012. « Approach for adaptive security monitor generation ». http://www.yumpu.com/en/document/view/8097461/approach-for-adaptive-security-monitor-generation-securechange.

Wiegers K.E. 2003. *Software requirements*. Microsoft press.

Wieringa R., Maiden N., Mead N., Rolland C. 2006. « Requirements engineering paper classification and evaluation criteria: a proposal and a discussion ». *Requirements Engineering* 11 (1): 102-7.

Wu Z., Palmer M. 1994. « Verbs semantics and lexical selection ». In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, 133-38. Association for Computational Linguistics.

Yin R. K. 2014. *Case study research: Design and methods*. Sage publications.

Yoshioka N., Washizaki H., Maruyama K. 2008. « A survey on security patterns ». *Progress in Informatics* 5 (5): 35-47.

Yu E., Liu L. 2001. « Modelling trust for system design using the i* strategic actors framework ». In *Trust in Cyber-societies*, 175-94. Springer.

Yu E., Strohmaier M., eDeng X. 2006. « Exploring Intentional Modeling and Analysis for Enterprise Architecture ». In the 10th IEEE International Enterprise Distributed Object Computing Conference Workshops, 2006. 32-32. IEEE.

Zuccato, A., Daniels N., Jampathom C. 2011. « Service Security Requirement Profiles for Telecom: How Software Engineers May Tackle Security ». In the *Sixth International Conference on Availability, Reliability and Security (ARES'11)*, 521-26. doi:10.1109/ARES.2011.81.